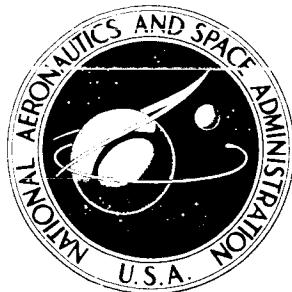


NASA CONTRACTOR
REPORT

NASA CR-909



NASA CR-909

A GEOMETRICALLY NONLINEAR
ANALYSIS OF ARBITRARILY
LOADED SHELLS OF REVOLUTION

by R. E. Ball

Prepared by
DYNAMIC SCIENCE
Monrovia, Calif.
for Langley Research Center

FACILITY FORM 602	ACCESSION NUMBER	STABIL
	PAGES	CODE
NASA CR OR TMX OR AD NUMBER		CATEGORY

NASA CR-909

A GEOMETRICALLY NONLINEAR ANALYSIS OF ARBITRARILY
LOADED SHELLS OF REVOLUTION

By R. E. Ball

Distribution of this report is provided in the interest of information exchange. Responsibility for the contents resides in the author or organization that prepared it.

Prepared under Contract No. NAS 1-5804 by
DYNAMIC SCIENCE
Monrovia, Calif.

for Langley Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 - CFSTI price \$3.00

CONTENTS

	PAGE
CONTENTS	iii and iv
SUMMARY	1
INTRODUCTION	2
SYMBOLS	4
SHELL GEOMETRY	8
THEORY	10
Strain-displacement Relations	10
Equilibrium Equations	11
Constitutive Relations	13
Boundary Conditions	13
SEPARATION OF VARIABLES	14
Fourier Expansions	14
Uncoupled Ordinary Differential Equations	18
Final Equations	22
Boundary Expressions	23
SOLUTION OF EQUATIONS	25
Finite Difference Formulation	25
Gauss Elimination	26
Singular Points	29
Load-deflection History	29
EXAMPLES	30
Circular Cylinder	30
Asymmetrically Loaded Spherical Cap	30
CONCLUSIONS	35
APPENDIX A -- SANDERS' NONLINEAR EQUATIONS FOR THE GENERAL SHELL	39
APPENDIX B -- NONLINEAR TERMS	42
APPENDIX C -- MATRIX EXPRESSIONS	46
APPENDIX D -- POLES	54
APPENDIX E -- PROGRAM WRITE-UP	62
Input Requirements	63
Optional Programming Requirements	73
Output Data	76
Subroutine Descriptions	79
APPENDIX F -- PROGRAM LISTING	90
APPENDIX G -- FLOW CHARTS	153
REFERENCES	210

FIGURES

- | | |
|--|----|
| 1. Shell Geometry and Coordinates | 9 |
| 2. Positive Directions for Displacements and Rotations | 12 |
| 3. Positive Directions for Forces, Moments and Loads | 12 |

PRECEDING PAGE BLANK NOT FILMED.

4. Matrix Equation for n = N	27
5. Deflection vs. Length for a Cylinder with U=V=W=0 and $(1 - \nu^2) M_s / Eh_o^2 = 10^{-2}$ at Both Ends	31
6. Moment vs. Length for a Cylinder with U=V=W=0 and $(1 - \nu^2) M_s / Eh_o^2 = 10^{-2}$ at Both Ends	32
7. Deflection vs. Length for a Cylinder with U=V=W=0 and $(1 - \nu^2) M_s / Eh_o^2 = 10^{-2} \cos 2\theta$ at Both Ends	33
8. Moment vs. Length for a Cylinder with U=V=W=0 and $(1 - \nu^2) M_s / Eh_o^2 = 10^{-2} \cos 2\theta$ at Both Ends	34
9. Load-deflection Curve for Half Loaded Shell	36
10. Deflection Profiles Along the 0 - 180° Meridian for Half Loaded Shell	37
11. Normal Displacement for Each Value of n at Station 3 of Meridian Centered Under Load	38
12. Input Format	64
13. FORTRAN Statements for Subroutine GEOM	74
14. Sample Output	80

TABLES

1. Important FORTRAN Variables	211
--	-----

A GEOMETRICALLY NONLINEAR ANALYSIS OF ARBITRARILY
LOADED SHELLS OF REVOLUTION

By R. E. Ball

SUMMARY

A digital computer program for the geometrically nonlinear analysis of thin elastic shells of revolution subjected to arbitrary load and temperature distributions has been developed to predict snap buckling of spacecraft shell structures due to asymmetric loads such as those imposed by reentry flow or landing impact. The analysis is based upon Sanders' nonlinear shell theory for the condition of small strains and moderately small rotations. The nonlinear partial differential equations are reduced to a sequence of linearized uncoupled ordinary differential equations by expanding the variables into Fourier series in the circumferential direction, and treating the nonlinear terms as pseudo loads. The linearized equations are repeatedly solved, using a finite difference formulation in conjunction with a Gaussian elimination procedure, until the solution converges.

All physical and geometrical properties of the shell are assumed to be axisymmetric, but the load and temperature distributions may be any function symmetric about a meridian and expressible by a Fourier series in the circumferential direction. The shell wall construction may be a composite with Young's modulus varying through its thickness. The thickness of the shell wall, the elastic modulus of the wall material, the applied load, and the thermal gradient through the wall may be smooth functions of the meridional arc length. The boundaries of the shell may be closed, free, fixed, or elastically restrained.

The program is coded in the FORTRAN IV language for operation in the IBSYS - IBJOB operating system (version 13). It has been dimensioned to allow a maximum of 20 equally spaced stations along the shell meridian and 10 arbitrary Fourier terms. A load-deformation history is determined using a variable incremental load step to assure reasonable convergence of the iteration process as the load nears the snap-through value. The output from the program consists of all input data plus the shell displacements, forces, and moments at every station for each Fourier mode and at prescribed locations around the circumference for each load step.

INTRODUCTION

Numerical analysis of shell structures is a subject that has received considerable attention in recent years. However, almost all of this attention has been given to linear and nonlinear axisymmetric problems. Apparently, the only successful nonlinear analysis of an asymmetrically loaded shell was the one by Famili and Archer (ref. 1) who developed a computer program for the finite asymmetric deformation of shallow spherical shells, basing their analysis on the nonlinear theory of Vlasov and using a finite difference formulation in conjunction with an iterative process to solve the nonlinear partial differential equations. As a consequence of the small amount of study in this area, a large number of problems remain unsolved, including the snap-buckling of spacecraft shell structures subjected to asymmetric loads such as those imposed by reentry flow or landing impact. The computer program described in this report should provide a tool for solving some of these problems.

The field equations used in this analysis were derived by Sanders (ref. 2) for the condition of small strains and moderately small rotations. The method for solving these equations follows the procedure employed by Budiansky and Radkowsky (ref. 3) in their analysis of the unsymmetrical bending of shells of revolution using Sanders' linear theory*, wherein the pertinent variables are expanded into Fourier series in the circumferential direction**. The nonlinear terms, which are products of series since they involve the total value of each variable, are each expanded into a single trigonometric series, and coefficients of like trigonometric arguments are grouped. In contrast to the linear analysis, the resulting equations do not uncouple into separate sets of ordinary differential equations for each Fourier index because of the presence of the nonlinear terms. However, the equations can be decoupled if each of the Fourier coefficients of the nonlinear terms are treated as known quantities, or pseudo loads, and an iterative process is used to obtain the solution.

Briefly, the set of linearized equations are first solved for each Fourier coefficient of the actual load, plus an estimated pseudo load, using a finite difference formulation in conjunction with a Gaussian elimination procedure. Next, the nonlinear terms are calculated using the new solution and entered into the equations as a revised estimate of

*Budiansky and Radkowsky's linear shell analysis has also been used as the basis of a computer program by Schaeffer (ref. 4).

**The application of Fourier series to nonlinear shell problems is not new, having been used previously by a number of investigators for studying the buckling behavior of axially loaded circular cylinders (see Hoff, et. al., ref. 5, and Thurston and Freeland, ref. 6), and by Bushnell (ref. 7) and Bushnell and Madsen (ref. 8) to solve the axisymmetric spherical cap problem.

the pseudo loads. All the sets of linearized equations are then solved again. This procedure repeats until the solution converges. After each iteration there may be additional sets of linear equations to solve that contain no Fourier coefficient of the actual load, but which do contain coefficients of the pseudo load arising from the nonlinear terms.

In order to determine the apparent snap-buckling load a variable incremental load step routine is built into the computer program. When the number of iterations are small the load is incremented in equal prescribed steps. As the load nears the snap-through value, and the number of iterations exceed a prescribed maximum, the incremental load is reduced. Any number of increment reductions can be made. The load is continually increased until either the prescribed maximum number of load steps have been taken or the shell enters a position of neutral equilibrium and snap buckling appears imminent. Post buckling behavior cannot be determined by this program because of the method of solution employed.

The body of this report contains the mathematical development of the governing equations and a description of the method of solution. Sanders' (ref. 2) nonlinear general shell field equations for his moderately small rotation theory are reproduced in Appendix A for the purpose of reference. The procedure for calculating the nonlinear terms is described in Appendix B. Appendix C contains the expressions for the coefficients of the unknowns in the governing equations, and Appendix D presents the "boundary" conditions for shells with a pole. A program write-up containing detailed flow charts, a description of the program usage with comprehensive input and output definitions and a test case are given in Appendix E. The source program is listed in Appendix F, and detailed flow charts are given in Appendix G.

The author takes this opportunity to acknowledge the significant contribution to this project by Jerry C. Peck.

SYMBOLS

a	=	reference length
B	=	inplane stiffness, equation (11a)
b	=	nondimensional inplane stiffness, equation (34a)
D	=	bending stiffness, equation (11b)
d	=	nondimensional bending stiffness, equation (34b)
E	=	Young's modulus
E_o	=	reference Young's modulus
$e_s, e_\theta, e_{s\theta}$	=	nondimensional Fourier coefficients for the reference surface strains, equations (21)
f_s	=	nondimensional Fourier coefficient for the transverse force, equations (25)
\hat{f}_s	=	nondimensional Fourier coefficient for the effective transverse force, equations (25)
h	=	thickness
h_o	=	reference thickness
i	=	Fourier index or meridian station
j	=	Fourier index
K	=	last meridian station
$k_s, k_\theta, k_{s\theta}$	=	nondimensional Fourier coefficients for the bending strains, equations (22)
$M_s, M_\theta, M_{s\theta}$	=	bending and twisting moments per unit length
$m_s, m_\theta, m_{s\theta}$	=	nondimensional Fourier coefficients for bending and twisting moments, equations (18)

SYMBOLS

m_T	=	nondimensional Fourier coefficient for the thermal bending moment, equation (34d)
$N_s, N_\theta, N_{s\theta}$	=	membrane forces per unit length
$\hat{N}_{s\theta}$	=	effective shear force, equation (13)
n	=	Fourier index
P	=	ratio of applied load to the classical buckling load of uniformly loaded sphere
$p_s, p_\theta, p_{s\theta}$	=	nondimensional Fourier coefficients for the components of the pressure load, equations (23)
Q_s, Q_θ	=	transverse forces per unit length
\hat{Q}_s	=	effective transverse force, equation (14)
q_s, q_θ, q	=	meridional, circumferential, and normal components of applied pressure load
R_s, R_θ	=	principal radii of curvature
r	=	normal distance from the axis of the shell
S	=	total length of meridian
s	=	meridional shell coordinate
T	=	local temperature change
$t_s, t_\theta, t_{s\theta}$	=	nondimensional Fourier coefficients for membrane forces, equations (17)
$\hat{t}_{s\theta}$	=	nondimensional Fourier coefficient for the effective shear force, equations (25)
t_T	=	nondimensional Fourier coefficient for the thermal membrane force, equation (34c)
U, V	=	displacements tangent to the meridian and to the parallel circle respectively

SYMBOLS

- u, v = nondimensional Fourier coefficients for the displacements tangent to the meridian and to the parallel circle respectively, equations (19)
- w = displacement normal to the reference surface
- w = nondimensional Fourier coefficient for the displacement normal to the reference surface, equations (19)
- α = coefficient of thermal expansion
- $\beta_s, \beta_\theta, \beta, \beta_{s\theta}$ = nondimensional coefficients for the nonlinear terms in the strain-displacement relations
- γ = ρ' / ρ
- Δ = nondimensional distance between stations
- $\epsilon_s, \epsilon_\theta, \epsilon_{s\theta}$ = reference surface strains, equations (5)
- ϵ_T = thermal membrane force, equation (11c)
- ζ = coordinate normal to the reference surface
- $\eta_{ss}, \eta_\theta, \eta_s$ = nondimensional coefficients for the nonlinear terms in the equilibrium equations
- $\eta_{s\theta}, \eta_{\theta\theta}, \eta_{s\theta}$
- θ = circumferential angle
- $\chi_s, \chi_\theta, \chi_{s\theta}$ = bending strains, equations (6)
- χ_T = thermal bending moment, equation (11d)
- λ = h_o / a
- $\mu_s^c, \eta_{sc}^c, \mu^s, \eta, \mu$ = coefficients, equations (B-7, B-9, and B-13)
- ν = Poisson's ratio
- ξ = nondimensional meridional coordinate, s/a
- ρ = nondimensional radius, r/a

SYMBOLS

σ_o	=	reference stress level
ω_s, ω_θ	=	nondimensional curvatures, $a/R_s, a/R_\theta$
$\Phi_s, \Phi_\theta, \Phi$	=	reference surface rotations, equations (7)
$\varphi_s, \varphi_\theta, \varphi$	=	nondimensional Fourier coefficients for the rotations, equations (20)
τ	=	Fourier coefficient for the local temperature change, equation (24)

MATRICES

A, B, C, P	=	4x4 matrices
E, F, G, H, J	=	4x4 matrices, Appendix C
e, f	=	1x4 matrices, Appendix C
q, x	=	1 x 4 column matrices
ℓ	=	1x4 boundary condition matrix
y	=	1x4 column matrix containing the boundary variables $t_s, \hat{t}_{s\theta}, \hat{f}_s, \text{ and } \varphi_s$
z	=	1x4 column matrix containing the unknowns $u, v, w, \text{ and } m_s$
Ω, Λ	=	4x4 boundary condition matrices
$\bar{\Omega}, \bar{\Lambda}$	=	4x4 boundary condition matrices

SHELL GEOMETRY

Consider the general shell of revolution shown in figure 1. Located within this shell is a reference surface. All material points of the shell can be located using the orthogonal coordinate system s, θ, ζ , where s is the meridional distance along the reference surface measured from one boundary, θ is the circumferential angle measured from a datum meridian plane, and ζ is the distance along the normal to the reference surface measured from the reference surface. The positive direction of each coordinate is indicated in figure 1. For convenience, let the reference surface be positioned so that

$$\int \zeta E d\zeta = 0 \quad (1)$$

where E is the elastic modulus and the integration is carried out through the thickness of the shell. Thus, when E is independent of ζ the reference surface coincides with the middle surface of the shell. Further, let the location of the reference surface be described by the dependent variable r , the normal distance from the axis of the shell. Accordingly, the principal radii of curvature of the reference surface are

$$\left. \begin{aligned} R_\theta &= r / \left[1 - (r')^2 \right]^{1/2} \\ R_s &= - \left[1 - (r')^2 \right]^{1/2} / r'' \end{aligned} \right\} \quad (2)$$

where a prime denotes differentiation with respect to s . Further, note the Codazzi identity

$$R'_\theta = r' \left(R_s^{-1} - R_\theta^{-1} \right) / r \quad (3)$$

and the relation

$$r''' = - r / R_s R_\theta \quad (4)$$

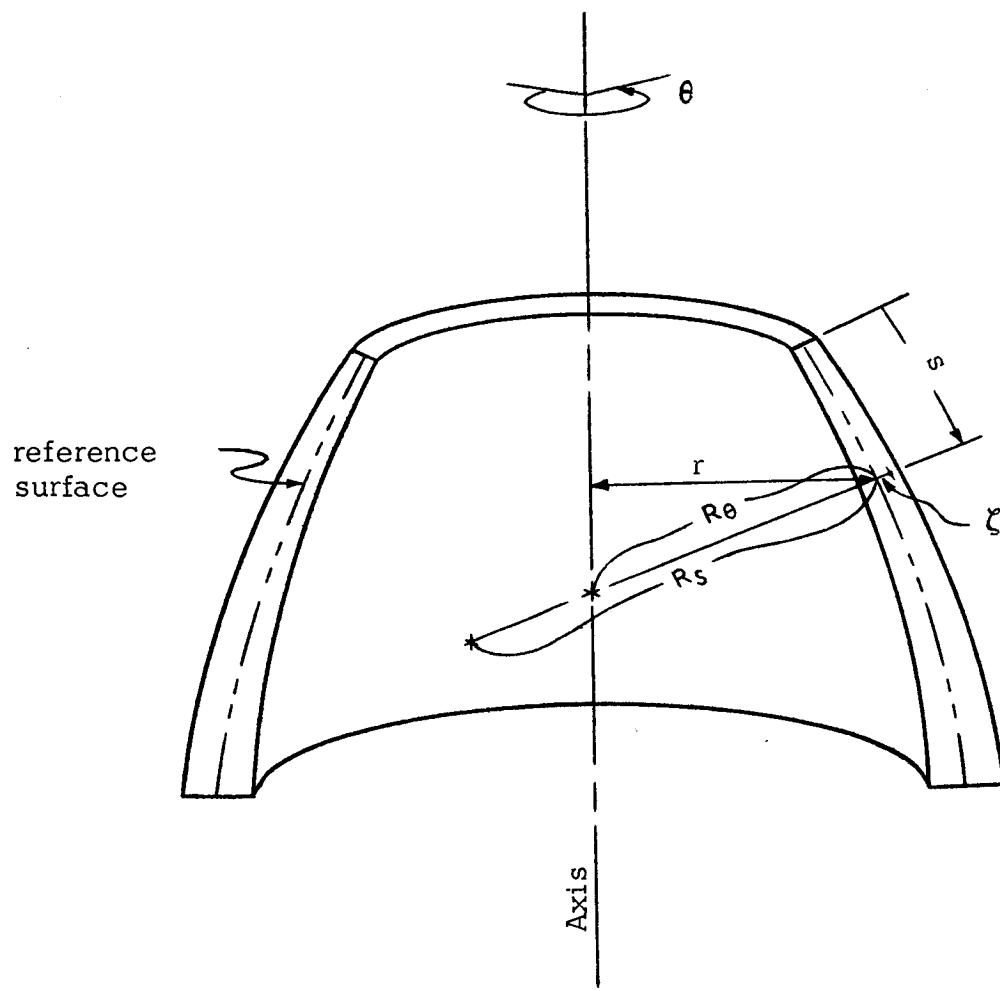


Figure 1. Shell Geometry and Coordinates

THEORY

Strain-displacement Relations

For a shell of revolution, the strain-displacement relations derived by Sanders take the form

$$\left. \begin{aligned} \epsilon_s &= U' + W/R_s + (\Phi_s^2 + \Phi^2)/2 \\ \epsilon_\theta &= V'/r + r' U/r + W/R_\theta + (\Phi_\theta^2 + \Phi^2)/2 \\ \epsilon_{s\theta} &= (V' + U'/r - r' V/r + \Phi_s \Phi_\theta)/2 \end{aligned} \right\} \quad (5)$$

and

$$\left. \begin{aligned} \chi_s &= \Phi_s' \\ \chi_\theta &= \Phi_\theta'/r + r' \Phi_s/r \\ \chi_{s\theta} &= [\Phi_\theta' + \Phi_s'/r - r' \Phi_\theta/r + (R_\theta^{-1} - R_s^{-1}) \Phi] / 2 \end{aligned} \right\} \quad (6)$$

where ϵ_s , ϵ_θ , and $\epsilon_{s\theta}$ are the reference surface strains, χ_s , χ_θ , and $\chi_{s\theta}$ are the bending strains, U and V are the displacements in the directions tangent to the meridian and to the parallel circle respectively, W is the displacement normal to the reference surface, and Φ_s , Φ_θ , and Φ are rotations defined by

$$\left. \begin{aligned} \Phi_s &= -W' + U/R_s \\ \Phi_\theta &= -W'/r + V/R_\theta \\ \Phi &= (V' + r' V/r - U'/r)/2 \end{aligned} \right\} \quad (7)$$

In these equations, and henceforth, dots denote differentiation with respect to θ . The positive direction of each displacement and rotation variable is indicated in figure 2.

Equilibrium Equations

Converting Sanders' equations for equilibrium to the s, θ coordinate system gives

$$\left. \begin{aligned} & \left(rN_s \right)' + N_{s\theta}' - r'N_\theta + rQ_s/R_s + \left(R_s^{-1} - R_\theta^{-1} \right) M_{s\theta}' / 2 \\ & = - rq_s + r \left(\Phi_s N_s + \Phi_\theta N_{s\theta} \right) / R_s + \left[\Phi \left(N_s + N_\theta \right) \right]' / 2 \\ & N_\theta' + \left(rN_{s\theta} \right)' + r'N_{s\theta} + rQ_\theta / R_\theta + r \left[\left(R_\theta^{-1} - R_s^{-1} \right) M_{s\theta} \right]' / 2 \\ & = - rq_\theta + r \left(\Phi_\theta N_\theta + \Phi_s N_{s\theta} \right) / R_\theta - r \left[\Phi \left(N_s + N_\theta \right) \right]' / 2 \\ & \left(rQ_s \right)' + Q_\theta' - rN_s / R_s - rN_\theta / R_\theta \\ & = - rq + \left(r \Phi_s N_s + r \Phi_\theta N_{s\theta} \right)' + \left(\Phi_s N_{s\theta} + \Phi_\theta N_\theta \right)' \end{aligned} \right\} \quad (8)$$

and

$$\left(rM_s \right)' + M_{s\theta}' - r' M_\theta - rQ_s = 0 \quad (9a)$$

$$M_\theta' + \left(rM_{s\theta} \right)' + r'M_{s\theta} - rQ_\theta = 0 \quad (9b)$$

where q_s , q_θ , and q are the meridional, circumferential, and normal components of the applied pressure load, Q_s and Q_θ are the transverse forces, N_s , N_θ and $N_{s\theta}$ are the membrane forces, and M_s , M_θ , and $M_{s\theta}$ are the bending and twisting moments. Refer to figure 3 for the positive directions of these variables.

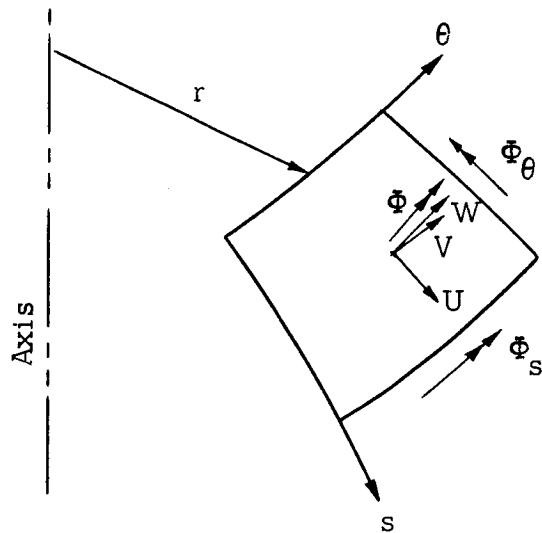


Figure 2. Positive Directions for Displacements and Rotations

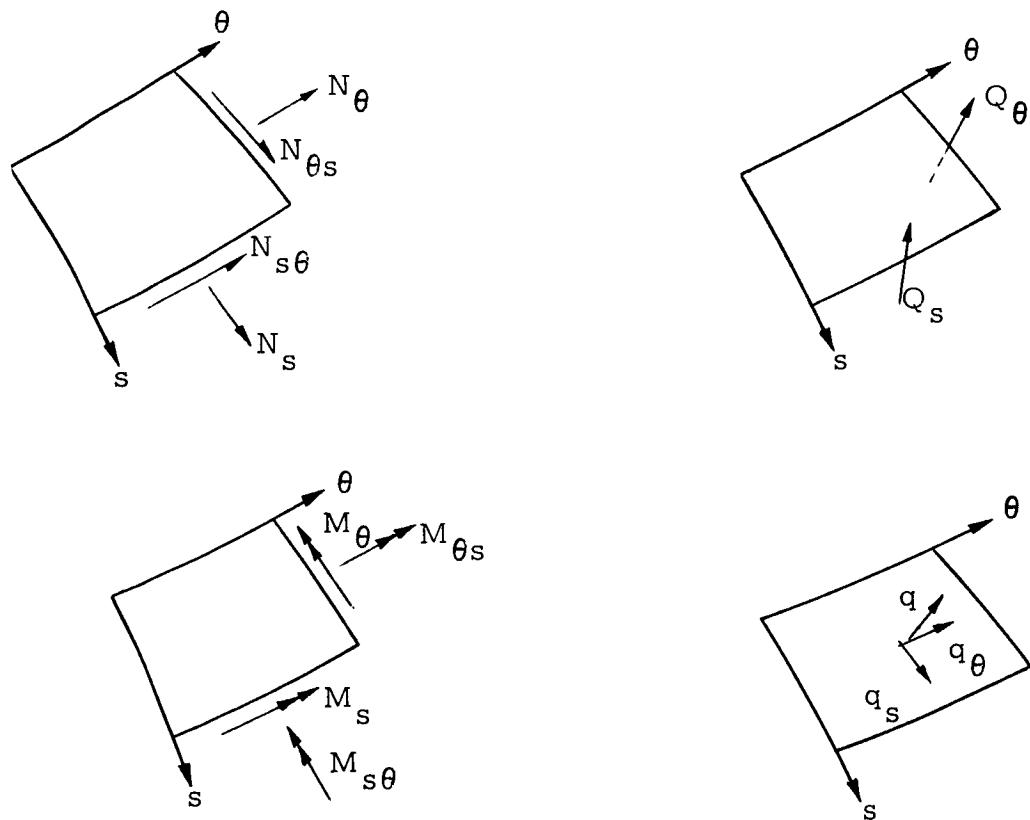


Figure 3. Positive Directions for Forces, Moments and Loads

Constitutive Relations

The constitutive relations used in Sanders' nonlinear theory are the same as those proposed by Love in his first approximation to the linear small strain theory of thin elastic shells. Noting equation (l), these can be given in the form

$$\left. \begin{aligned}
 B \epsilon_s &= N_s - \nu N_\theta + \epsilon_T \\
 B \epsilon_\theta &= N_\theta - \nu N_s + \epsilon_T \\
 B \epsilon_{s\theta} &= (1 + \nu) N_{s\theta} \\
 D\kappa_s &= M_s - \nu M_\theta + \kappa_T \\
 D\kappa_\theta &= M_\theta - \nu M_s + \kappa_T \\
 D\kappa_{s\theta} &= (1 + \nu) M_{s\theta}
 \end{aligned} \right\} \quad (10)$$

where ν is Poisson's ratio, assumed constant through the thickness, and

$$B = \int E d \zeta \quad (11a)$$

$$D = \int \zeta^2 E d \zeta \quad (11b)$$

$$\epsilon_T = \int \alpha T E d \zeta \quad (11c)$$

$$\kappa_T = \int \zeta \alpha T E d \zeta \quad (11d)$$

T is the local temperature change, and α is the coefficient of thermal expansion.

Boundary Conditions

In Sanders' nonlinear theory, the conditions to prescribe on the edges of a shell of revolution are

$$\left. \begin{aligned}
 N_s \text{ or } U &\quad \hat{N}_{s\theta} \text{ or } V \\
 \hat{Q}_s \text{ or } W &\quad M_s \text{ or } \Phi_s
 \end{aligned} \right\} \quad (12)$$

where $\hat{N}_{s\theta}$ and \hat{Q}_s are effective shear and transverse forces defined by

$$\hat{N}_{s\theta} = N_{s\theta} + \left(3 R_\theta^{-1} - R_s^{-1} \right) M_{s\theta} / 2 + (N_s + N_\theta) \Phi / 2 \quad (13)$$

$$\hat{Q}_s = Q_s + M_{s\theta} / r - \Phi_s N_s - \Phi_\theta N_{s\theta} \quad (14)$$

Using equilibrium equation (9a) to eliminate Q_s from equation (14) leads to

$$\hat{Q}_s = \left[\left(r M_s \right)' + 2 M_{s\theta}' - r' M_\theta \right] / r - \Phi_s N_s - \Phi_\theta N_{s\theta} \quad (15)$$

Elastic restraints at the edge of a shell can be provided for by linearly relating the forces or moment to the appropriate displacements or rotation. Consequently, the boundary conditions may be given in the matrix form

$$\bar{\Omega} \begin{bmatrix} N_s \\ \hat{N}_{s\theta} \\ \hat{Q}_s \\ \Phi_s \end{bmatrix} + \bar{\Lambda} \begin{bmatrix} U \\ V \\ W \\ M_s \end{bmatrix} = \ell \quad (16)$$

where $\bar{\Omega}$ and $\bar{\Lambda}$ are 4×4 matrices and ℓ is a column matrix. The values of the elements of these matrices are determined by the conditions prescribed at the shell boundary.

SEPARATION OF VARIABLES

Fourier Expansions

The crux of the method used here to solve the nonlinear field equations is the elimination of the independent variable θ by expanding all dependent variables into Fourier sine or cosine series in the circumferential direction. Only loading conditions that are symmetric about a datum meridian plane will be considered.

Thus, the variables can be expressed as follows*:

$$\left. \begin{aligned} N_s &= \sigma_o h_o \sum_{n=0}^{\infty} t_s^{(n)} \cos n\theta \\ N_\theta &= \sigma_o h_o \sum_{n=0}^{\infty} t_\theta^{(n)} \cos n\theta \\ N_{s\theta} &= \sigma_o h_o \sum_{n=1}^{\infty} t_{s\theta}^{(n)} \sin n\theta \end{aligned} \right\} \quad (17)$$

$$\left. \begin{aligned} M_s &= \frac{\sigma_o h_o^3}{a} \sum_{n=0}^{\infty} m_s^{(n)} \cos n\theta \\ M_\theta &= \frac{\sigma_o h_o^3}{a} \sum_{n=0}^{\infty} m_\theta^{(n)} \cos n\theta \\ M_{s\theta} &= \frac{\sigma_o h_o^3}{a} \sum_{n=1}^{\infty} m_{s\theta}^{(n)} \sin n\theta \end{aligned} \right\} \quad (18)$$

*Theoretically, the complete Fourier series including both the sine and cosine expansions should be used because of the possibility of "odd" displacements occurring under "even" loads, i. e., a bifurcation phenomenon. This aspect is not considered in the program.

$$\left. \begin{aligned}
 U &= \frac{a \sigma_o}{E_o} \sum_{n=0}^{\infty} u^{(n)} \cos n\theta \\
 V &= \frac{a \sigma_o}{E_o} \sum_{n=1}^{\infty} v^{(n)} \sin n\theta \\
 W &= \frac{a \sigma_o}{E_o} \sum_{n=0}^{\infty} w^{(n)} \cos n\theta
 \end{aligned} \right\} \quad (19)$$

$$\left. \begin{aligned}
 \Phi_s &= \frac{\sigma_o}{E_o} \sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \\
 \Phi_\theta &= \frac{\sigma_o}{E_o} \sum_{n=1}^{\infty} \varphi_\theta^{(n)} \sin n\theta \\
 \Phi &= \frac{\sigma_o}{E_o} \sum_{n=1}^{\infty} \varphi^{(n)} \sin n\theta
 \end{aligned} \right\} \quad (20)$$

$$\left. \begin{aligned}
 \epsilon_s &= \frac{\sigma_o}{E_o} \sum_{n=0}^{\infty} e_s^{(n)} \cos n\theta \\
 \epsilon_\theta &= \frac{\sigma_o}{E_o} \sum_{n=0}^{\infty} e_\theta^{(n)} \cos n\theta \\
 \epsilon_{s\theta} &= \frac{\sigma_o}{E_o} \sum_{n=1}^{\infty} e_{s\theta}^{(n)} \sin n\theta
 \end{aligned} \right\} \quad (21)$$

$$\left. \begin{aligned}
 x_s &= \frac{\sigma_o}{aE_o} \sum_{n=0}^{\infty} k_s^{(n)} \cos n\theta \\
 x_\theta &= \frac{\sigma_o}{aE_o} \sum_{n=0}^{\infty} k_\theta^{(n)} \cos n\theta \\
 x_{s\theta} &= \frac{\sigma_o}{aE_o} \sum_{n=1}^{\infty} k_{s\theta}^{(n)} \sin n\theta
 \end{aligned} \right\} \quad (22)$$

$$\left. \begin{aligned}
 q_s &= \frac{\sigma_o h_o}{a} \sum_{n=0}^{\infty} p_s^{(n)} \cos n\theta \\
 q_\theta &= \frac{\sigma_o h_o}{a} \sum_{n=1}^{\infty} p_\theta^{(n)} \sin n\theta \\
 q &= \frac{\sigma_o h_o}{a} \sum_{n=0}^{\infty} p^{(n)} \cos n\theta
 \end{aligned} \right\} \quad (23)$$

$$T = \sum_{n=0}^{\infty} \tau^{(n)} \cos n\theta \quad (24)$$

$$\left. \begin{aligned}
 \hat{N}_{s\theta} &= \sigma_o h_o \sum_{n=1}^{\infty} \hat{t}_{s\theta}^{(n)} \sin n\theta \\
 \hat{Q}_s &= \sigma_o h_o \sum_{n=0}^{\infty} \hat{f}_s^{(n)} \cos n\theta \\
 Q_s &= \sigma_o h_o \sum_{n=0}^{\infty} f_s^{(n)} \cos n\theta
 \end{aligned} \right\} \quad (25)$$

where the Fourier coefficients have been nondimensionalized and "normalized" with a reference stress level σ_o , Young's modulus E_o , thickness h_o , and length a , to reduce round off error in the computations. Henceforth, for convenience, the superscript n on the Fourier coefficients will be dropped. It should be noted that although an infinite series is indicated in equations (17-25) only ten terms can actually be handled in the program.

Uncoupled Ordinary Differential Equations

In order to eliminate the independent variable θ from the problem, and convert the partial differential equations to ordinary uncoupled differential equations, the nonlinear terms require special treatment. Every nonlinear term is the product of two Fourier series; however, each product can be reduced to a single trigonometric series wherein the coefficient is itself a series. Accordingly, it can be shown that substituting equations (20) into the nonlinear terms contained in equations (5) will lead to

$$\left. \begin{aligned} \Phi_s^2 &= \left(\frac{\sigma_o}{E_o} \right)^2 \left(\sum_{n=0}^{\infty} \varphi_s \cos n\theta \right)^2 = \frac{\sigma_o}{E_o} \sum_{n=0}^{\infty} \beta_s \cos n\theta \\ \Phi_\theta^2 &= \left(\frac{\sigma_o}{E_o} \right)^2 \left(\sum_{n=1}^{\infty} \varphi_\theta \sin n\theta \right)^2 = \frac{\sigma_o}{E_o} \sum_{n=0}^{\infty} \beta_\theta \cos n\theta \\ \Phi^2 &= \left(\frac{\sigma_o}{E_o} \right)^2 \left(\sum_{n=1}^{\infty} \varphi \sin n\theta \right)^2 = \frac{\sigma_o}{E_o} \sum_{n=0}^{\infty} \beta \cos n\theta \\ \Phi_s \Phi_\theta &= \left(\frac{\sigma_o}{E_o} \right)^2 \left(\sum_{n=0}^{\infty} \varphi_s \cos n\theta \right) \left(\sum_{n=1}^{\infty} \varphi_\theta \sin n\theta \right) \\ &\quad = \frac{\sigma_o}{E_o} \sum_{n=1}^{\infty} \beta_{s\theta} \sin n\theta \end{aligned} \right\} \quad (26)$$

In this analysis, the β 's are assumed to be known quantities, and the procedure for calculating them is described in Appendix B. Therefore, substituting equations (26) and the appropriate Fourier series into equations (5-7) yields the nondimensional uncoupled

ordinary differential equations for the strain-displacement relations

$$\left. \begin{aligned} e_s &= u' + \omega_s w + (\beta_s + \beta)/2 \\ e_\theta &= nv/\rho + \gamma u + \omega_\theta w + (\beta_\theta + \beta)/2 \\ e_{s\theta} &= (v' - nu/\rho - \gamma v + \beta_{s\theta})/2 \end{aligned} \right\} \quad (27)$$

$$\left. \begin{aligned} k_s &= \varphi_s' \\ k_\theta &= n \varphi_\theta / \rho + \gamma \varphi_s \\ k_{s\theta} &= [\varphi_\theta' - n \varphi_s / \rho - \gamma \varphi_\theta + (\omega_\theta - \omega_s) \varphi] / 2 \end{aligned} \right\} \quad (28)$$

$$\left. \begin{aligned} \varphi_s &= -w' + \omega_s u \\ \varphi_\theta &= nw/\rho + \omega_\theta v \\ \varphi &= (v' + \gamma v + nu/\rho)/2 \end{aligned} \right\} \quad (29)$$

where $\rho = r/a$, $\gamma = p'/\rho$, $\omega_\theta = a/R_\theta$, $\omega_s = a/R_s$. In these and the following nondimensional equations, primes denote differentiation with respect to the nondimensional coordinate $\xi = s/a$.

The equilibrium equations can be converted to nondimensional uncoupled ordinary differential equations in a similar manner. Using equations (9a and 9b) to eliminate Q_s and Q_θ from equations (8), substituting the appropriate series expansions into the results, and using the geometrical identities given in equations (3 and 4) leads to

$$\begin{aligned}
& t'_s + \gamma(t_s - t_\theta) + n t_{s\theta} / \rho + \lambda^2 \left[\omega_s m'_s + \gamma \omega_s (m_s - m_\theta) \right. \\
& \left. + n(3\omega_s - \omega_\theta)m_{s\theta} / 2\rho \right] = -p_s + \omega_s (\eta_{ss} + \eta_{\theta s}) \\
& + n(\eta_s + \eta_\theta) / (2\rho) \\
\\
& t'_{s\theta} + 2\gamma t_{s\theta} - n t_\theta / \rho + \lambda^2 \left[-n \omega_\theta m_\theta / \rho + (3\omega_\theta - \omega_s)m'_{s\theta} / 2 \right. \\
& \left. + \gamma(3\omega_\theta + \omega_s)m_{s\theta} / 2 - \omega'_s m_{s\theta} / 2 \right] = -p_\theta + \omega_\theta (\eta_{\theta\theta} + \eta_{s\theta}) \\
& - (\eta_s + \eta_\theta)' / 2 \\
\\
& - \omega_s t_s - \omega_\theta t_\theta + \lambda^2 \left[m'_s + 2\gamma m'_s - \omega_s \omega_\theta m_s + (\omega_s \omega_\theta - n^2 / \rho^2)m_\theta \right. \\
& \left. - \gamma m'_\theta + 2nm'_{s\theta} / \rho + 2\gamma nm_{s\theta} / \rho \right] = -p + (\rho \eta_{ss} + \rho \eta_{\theta s})' / \rho \\
& + n(\eta_{s\theta} + \eta_{\theta\theta}) / \rho
\end{aligned} \tag{30}$$

where $\lambda = h_o/a$ and the η terms are taken as known quantities defined by

$$\begin{aligned}
\Phi_s N_s &= \frac{\sigma_o^2 h_o}{E_o} \left(\sum_{n=0}^{\infty} \varphi_s \cos n\theta \right) \left(\sum_{n=0}^{\infty} t_s \cos n\theta \right) \\
&= \sigma_o h_o \sum_{n=0}^{\infty} \eta_{ss} \cos n\theta
\end{aligned}$$

$$\Phi_{\theta} N_s \theta = \frac{\sigma_o^2 h_o}{E_o} \left(\sum_{n=1}^{\infty} \varphi_{\theta} \sin n\theta \right) \left(\sum_{n=1}^{\infty} t_{s\theta} \sin n\theta \right)$$

$$= \sigma_o h_o \sum_{n=0}^{\infty} \eta_{\theta s} \cos n\theta$$

$$\Phi N_s = \frac{\sigma_o^2 h_o}{E_o} \left(\sum_{n=1}^{\infty} \varphi \sin n\theta \right) \left(\sum_{n=0}^{\infty} t_s \cos n\theta \right)$$

$$= \sigma_o h_o \sum_{n=1}^{\infty} \eta_s \sin n\theta$$

$$\Phi N_{\theta} = \frac{\sigma_o^2 h_o}{E_o} \left(\sum_{n=1}^{\infty} \varphi \sin n\theta \right) \left(\sum_{n=0}^{\infty} t_{\theta} \cos n\theta \right)$$

$$= \sigma_o h_o \sum_{n=1}^{\infty} \eta_{\theta} \sin n\theta$$

$$\Phi_{\theta} N_{\theta} = \frac{\sigma_o^2 h_o}{E_o} \left(\sum_{n=1}^{\infty} \varphi_{\theta} \sin n\theta \right) \left(\sum_{n=0}^{\infty} t_{\theta} \cos n\theta \right)$$

$$= \sigma_o h_o \sum_{n=1}^{\infty} \eta_{\theta \theta} \sin n\theta$$

$$\Phi_s N_{s\theta} = \frac{\sigma_o^2 h_o}{E_o} \left(\sum_{n=0}^{\infty} \varphi_s \cos n\theta \right) \left(\sum_{n=1}^{\infty} t_{s\theta} \sin n\theta \right)$$

$$= \sigma_o h_o \sum_{n=1}^{\infty} \eta_{s\theta} \sin n\theta$$

(31)

The procedure for calculating the η 's is described in Appendix B.

Finally, since equations (10) are linear, a simple inversion leads to

$$\left. \begin{aligned} t_s &= b(e_s + \nu e_\theta) - t_T \\ t_\theta &= b(e_\theta + \nu e_s) - t_T \\ t_{s\theta} &= b(1-\nu) e_{s\theta} \end{aligned} \right\} \quad (32)$$

and

$$m_s = d(k_s + \nu k_\theta) - m_T \quad (33a)$$

$$m_\theta = d(k_\theta + \nu k_s) - m_T \quad (33b)$$

$$m_{s\theta} = d(1-\nu) k_{s\theta} \quad (33c)$$

where

$$b = \int E d\xi / [E_0 h_0 (1-\nu^2)] \quad (34a)$$

$$d = \int \xi^2 E d\xi / [E_0 h_0^3 (1-\nu^2)] \quad (34b)$$

$$t_T = \int \alpha \tau E d\xi / [\sigma_0 h_0 (1-\nu)] \quad (34c)$$

$$m_T = a \int \xi \alpha \tau E d\xi / [\sigma_0 h_0^3 (1-\nu)] \quad (34d)$$

Final Equations

Budiansky and Radkowsky (ref. 3) have shown that, for the linear shell problem, the set of uncoupled field equations can be reduced to four second order differential equations in terms of

the four unknowns u , v , w , and m_s , provided the bending moment m_θ is replaced by the equality

$$m_\theta = \nu m_s + d(1 - \nu^2) k_\theta - (1 - \nu) m_T \quad (35)$$

to prevent derivatives of w higher than two from appearing. The same procedure is used here.

Three of the final four equations are derived from the equilibrium equations (30) by applying equations (32, 33c, and 35) with the membrane and bending strains expressed in terms of the displacements according to equations (27-29). The fourth equation is the meridional bending moment-curvature relationship given by equation (33a), with k_s and k_θ in terms of the displacements. A convenient representation of these four equations is the matrix form

$$E z'' + F z' + G z = e \quad (36)$$

where

$$z = \begin{bmatrix} u \\ v \\ w \\ m_s \end{bmatrix}$$

and E , F , G , and e are matrices defined in Appendix C. The elements of E , F , and G are identical with those given in reference 3, but the e matrix contains both the load terms of the linear theory and all of the nonlinear terms.

Boundary Expressions

The one set of boundary conditions that applies to every value of n is obtained by substituting the appropriate Fourier series into equations (13, 15, and 16), giving

$$\bar{\Omega} \begin{bmatrix} \sigma_0^h t_s \\ \hat{\sigma}_0^h t_{s\theta} \\ \hat{\sigma}_0^h f_s \\ \frac{\sigma_0}{E_0} \varphi_s \end{bmatrix} + \bar{\Lambda} \begin{bmatrix} \frac{a\sigma_0}{E_0} u \\ \frac{a\sigma_0}{E_0} v \\ \frac{a\sigma_0}{E_0} w \\ \frac{\sigma_0 h^3}{a} m_s \end{bmatrix} = \lambda \quad (37)$$

where

$$\begin{aligned} \hat{t}_{s\theta} &= t_{s\theta} + \lambda^2 (3\omega_\theta - \omega_s) m_{s\theta} / 2 + (\eta_s + \eta_\theta) / 2 \\ \hat{f}_s &= \lambda^2 \left[m_s' + \gamma (m_s - m_\theta) + 2n m_{s\theta} / \rho \right] - (\eta_{ss} + \eta_{\theta s}) \end{aligned} \quad (38)$$

By incorporating the dimensional constants with $\bar{\Omega}$ and $\bar{\Lambda}$, equation (37) can be expressed in the nondimensional form

$$\Omega y + \Lambda z = \lambda \quad (39)$$

where

$$y = \begin{bmatrix} t_s \\ \hat{t}_{s\theta} \\ \hat{f}_s \\ \varphi_s \end{bmatrix}$$

In order to complete the formulation of the boundary conditions, the boundary variables y must be expressed in terms of the unknowns z . Accordingly, it follows from equations (27-29, 32, 33c, 35, and 38) that

$$y = H z' + J z + f \quad (40)$$

where H , J and f are matrices defined in Appendix C. Matrices H and J are identical with those given in reference 3, but f contains the nonlinear terms. Finally, substituting equation (40) into equation (39) gives

$$\Omega H z' + (\Lambda + \Omega J) z = \ell - \Omega f \quad (41)$$

SOLUTION OF EQUATIONS

Finite Difference Formulation

Let the shell meridian be divided into $K - 1$ equal increments, and denote the end of each increment or station by the index i . Thus, $i = 1$ corresponds to the initial edge of the shell and $i = K$ corresponds to the final edge. Two fictitious stations off each end of the shell, $i = 0$ and $i = K + 1$, are introduced.

Let the first and second differentials of z at station i be approximated by

$$z'_i = (z_{i+1} - z_{i-1}) / 2 \Delta \quad (42a)$$

$$z''_i = (z_{i+1} - 2z_i + z_{i-1}) / \Delta^2 \quad (42b)$$

where Δ is the nondimensional distance between stations. Substituting equations (42a and 42b) into equation (36) leads to

$$A_i z_{i+1} + B_i z_i + C_i z_{i-1} = g_i \quad (43)$$

where

$$A_i = 2E_i / \Delta + F_i$$

$$B_i = -4E_i / \Delta + 2\Delta G_i$$

$$C_i = 2E_i / \Delta - F_i$$

$$g_i = 2\Delta e_i$$

and $i = 1, 2, \dots, K$ to insure equilibrium over the total length of the shell.

At the boundaries equation (41) must hold. Thus, substituting equation (42a) into equation (41) leads to

$$\Omega_1 \left\{ H_1 (z_2 - z_0) / 2 \Delta + J_1 z_1 + f_1 \right\} + \Lambda_1 z_1 = \ell_1 \quad (44a)$$

$$\Omega_K \left\{ H_K (z_{K+1} - z_{K-1}) / 2 \Delta + J_K z_K + f_K \right\} + \Lambda_K z_K = \ell_K \quad (44b)$$

Gauss Elimination

Equations (43, 44a, and 44b) constitute a set of algebraic equations. These equations can be expressed in the matrix form illustrated in figure 4 for the typical case where $n = N$. There is one of these matrix equations for every value of n considered. Each matrix equation is solved for the $z_i^{(n)}$ independently from all the others by a form of Gaussian elimination. Setting $i = 1$ in equation (43) and solving for z_0 gives*

$$z_0 = C_1^{-1} \left\{ g_1 - A_1 z_2 - B_1 z_1 \right\} \quad (45)$$

Substituting equation (45) into equation (44a) and solving for z_1 gives

$$\begin{aligned} z_1 &= \left[\Omega_1 \left\{ \frac{H_1}{2 \Delta} C_1^{-1} B_1 + J_1 \right\} + \Lambda_1 \right]^{-1} \left\{ - \frac{\Omega_1 H_1}{2 \Delta} (1 + C_1^{-1} A_1) z_2 \right. \\ &\quad \left. + \ell_1 + \Omega_1 \frac{H_1}{2 \Delta} C_1^{-1} g_1 - \Omega_1 f_1 \right\} \end{aligned} \quad (46)$$

Next, a general recursion relation for z_i in terms of z_{i+1} can be given in the form

$$z_i = -P_i z_{i+1} + x_i \quad (47)$$

where $i = 1, 2, \dots, K$. Therefore, according to equation (46)

*Equation (43) is used to obtain z_0 rather than equation (44a) because there is a possibility that $\Omega_1 = 0$.

$$\begin{bmatrix}
-\frac{\Omega_{11}^{H(N)}}{2\Delta} & \Omega_{11}^{J(N)} + \Lambda_1 & \frac{\Omega_{11}^{H(N)}}{2\Delta} \\
C_1^{(N)} & B_1^{(N)} & A_1^{(N)} \\
C_2^{(N)} & B_2^{(N)} & A_2^{(N)}
\end{bmatrix}
\begin{bmatrix}
Z_O^{(N)} \\
Z_1^{(N)} \\
Z_2^{(N)} \\
\vdots \\
Z_K^{(N)}
\end{bmatrix}
=
\begin{bmatrix}
\ell_1 - \Omega_{11}^{f(N)} \\
g_1^{(N)} \\
g_2^{(N)} \\
\vdots \\
g_K^{(N)}
\end{bmatrix}$$

$$\begin{bmatrix}
C_K^{(N)} & B_K^{(N)} & A_K^{(N)} \\
\frac{-\Omega_{KK}^{H(N)}}{2\Delta} & \Omega_{KK}^{J(N)} + \Lambda_K & \frac{\Omega_{KK}^{H(N)}}{2\Delta}
\end{bmatrix}
\begin{bmatrix}
Z_{K+1}^{(N)} \\
Z_K^{(N)}
\end{bmatrix}
= \ell_K - \Omega_{KK}^{f(N)}$$

FIGURE 4. Matrix Equation for $n = N$

$$P_i = \left[\Omega_1 \left\{ \frac{H_1}{2\Delta} C_1^{-1} B_1 + J_1 \right\} + A_1 \right]^{-1} \left\{ \Omega_1 \frac{H_1}{2\Delta} \left(1 + C_1^{-1} A_1 \right) \right\} \quad (48a)$$

and

$$x_i = \left[\Omega_1 \left\{ \frac{H_1}{2\Delta} C_1^{-1} B_1 + J_1 \right\} + A_1 \right]^{-1} \left\{ \ell_1 + \Omega_1 \left[\frac{H_1}{2\Delta} C_1^{-1} g_1 - f_1 \right] \right\} \quad (48b)$$

Substituting equation (47) into equation (43) gives the recursion expressions for P_i and x_i

$$P_i = \left[B_i - C_i P_{i-1} \right]^{-1} A_i \quad (49a)$$

and

$$x_i = \left[B_i - C_i P_{i-1} \right]^{-1} \left\{ g_i - C_i x_{i-1} \right\} \quad (49b)$$

Starting with equations (48a and 48b), P_i and x_i can be calculated from equations (49a and 49b) for $i = 2, \dots, K$.

At station $i = K - 1$ and $i = K$ equation (47) becomes

$$z_{K-1} = -P_{K-1} z_K + x_{K-1} \quad (50a)$$

$$z_K = -P_K z_{K+1} + x_K \quad (50b)$$

Substituting equations (50a and 50b) into equation (44b) and solving for z_{K+1} gives

$$\begin{aligned} z_{K+1} &= \left[\Omega_K \frac{H_K}{2\Delta} \left(1 - P_{K-1} P_K \right) - (\Omega_K J_K + \Lambda_K) P_K \right]^{-1} \left\{ \ell_K \right. \\ &\quad \left. + \Omega_K \frac{H_K}{2\Delta} \left(-P_{K-1} x_K + x_{K-1} \right) - (\Omega_K J_K + \Lambda_K) x_K - \Omega_K f_K \right\} \end{aligned} \quad (51)$$

With z_{K+1} given by equation (51), a backward sweep using equation (47) provides the solution for z_i for $1 \leq i \leq K$. The solution for the unknowns at $i=0$ is given by equation (45).

This particular method for solving equation (36) is an efficient one to use in a load history and an iteration procedure when only the load matrix g changes since only x needs to be recalculated.

Singular Points

The equations (48a and 48b) for P_1 and x_1 and equation (51) for z_{K+1} are applicable when the shell has edges. If the shell has a pole, $\rho=0$, and special "boundary" conditions are required. These conditions are derived and the associated matrices P_1 and x_1 and the solution for z_K are formulated in Appendix D.

Load-deflection History

The basic procedure for determining the load-deflection history of a shell subjected to a given design load is as follows:

- 1) A solution to equation (36) is obtained for a specified increment of each Fourier coefficient of the design load. All pseudo loads are taken as zero.
- 2) This solution is used to calculate the β 's and η 's, and a new value of the load vector g is obtained for each n . New values of n may be introduced by the nonlinear terms.
- 3) A solution to equation (36) is obtained for the new value of g for each n , and is compared with the previous solution.
- 4) If the difference between the two solutions is larger than a specified percentage then step #2 is repeated. However, if the number of iterations has exceeded a specified maximum, the total load is reduced by one increment, the increment is reduced by a factor of 5, and this new increment is added to the load. If a specified number of load reductions have been made, snap buckling is assumed to be imminent and the program ends.
- 5) If the solutions have converged, another load increment is added provided the number of load steps is less than a specified maximum. An estimate of the solution for this new load is made by linear extrapolation, and step #2 is repeated.

EXAMPLES

Circular Cylinder

A linear solution has been obtained from the computer program for the case of a uniform circular cylinder of constant thickness h_o and subjected to the end moments

$$M_s = 10^{-2} \left[E h_o / (1 - \nu^2) \right] \cos n\theta$$

with

$$U = V = W = 0$$

at both edges. The radius to thickness ratio was taken as 50 and the length to radius ratio was 1. Poisson's ratio was .3. Twenty stations along the meridian were used. Budiansky and Radkowsky (ref. 3) solved the same problem using 300 intervals over the length of the cylinder. Their results for W and M_s for $n = 0$ and 2 are presented in figures 5-8 for comparison with those from this program.

Asymmetrically Loaded Spherical Cap

In order to check the validity of the nonlinear solution for asymmetrically loaded shells, the problem of a clamped spherical cap subjected to a uniform pressure over one-half of the shell surface is studied. This problem was first considered by Famili and Archer (ref. 1) using the nonlinear theory of Vlasov* and the method of finite differences for solution. They analyzed a cap with the property

$$\{ [12(1 - \nu^2)]^{1/2} A^2 / R_s h \}^{1/2} = 6$$

where A is the maximum value of r. In the case executed using the analysis presented here, the meridian length $S = 105$ in., the radius of curvature $R_s = 1000$ in., the thickness $h = 1$ in., and Poisson's ratio $\nu = .3$. A uniform load $q = -30$ psi, was applied over the portion $\theta = -90^\circ$ to $\theta = 90^\circ$. In Famili and Archer's (ref. 1) analysis, 7 stations along the meridian and 6 meridians were used.

Famili and Archer's (ref. 1) results for the load-deflection history of a station located near the apex of the shell are given in

* Refer to Appendix A for a comparison of Vlasov's theory with that of Sanders.

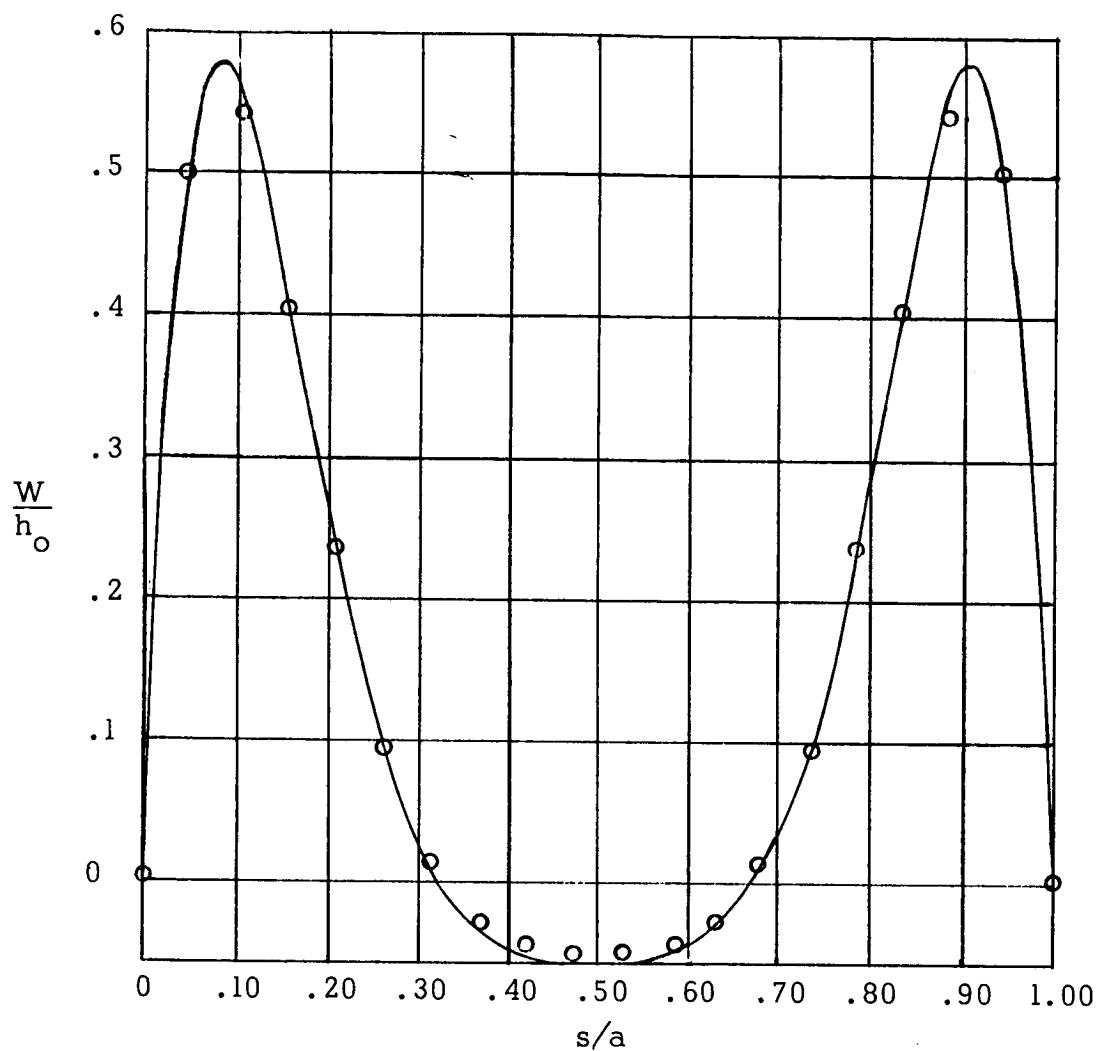


Figure 5. Deflection vs. Length for a Cylinder with $U=V=W=0$ and
 $(1 - \nu^2) M_s / Eh_o^2 = 10^{-2}$ at Both Ends

— Reference 3, 300 points

○ This program, 20 points

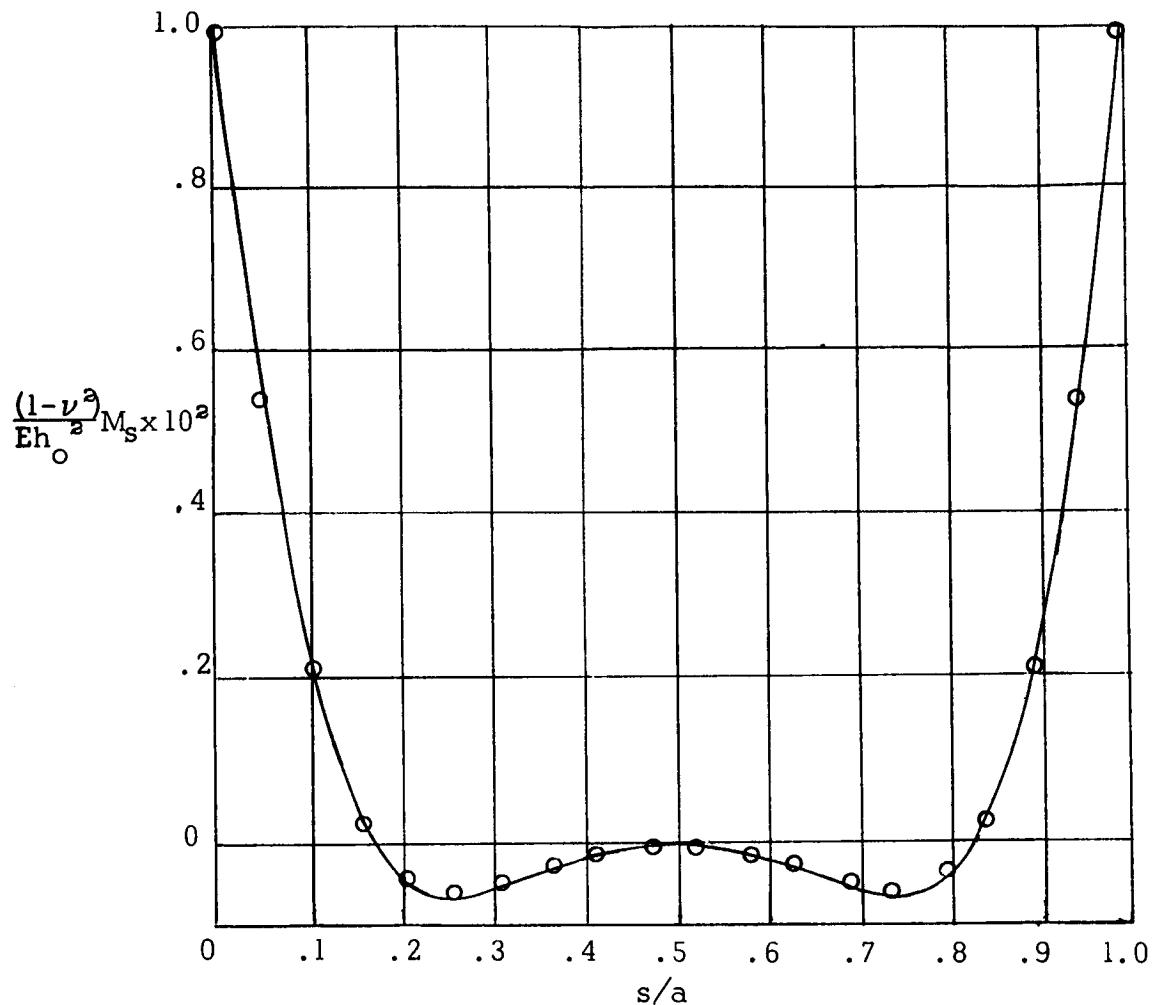


Figure 6. Moment vs. Length for a Cylinder with $U=V=W=0$ and
 $(1-\nu^2)M_s/Eh_0^2 = 10^{-2}$ at Both Ends

— Reference 3, 300 points

○ This program, 20 points

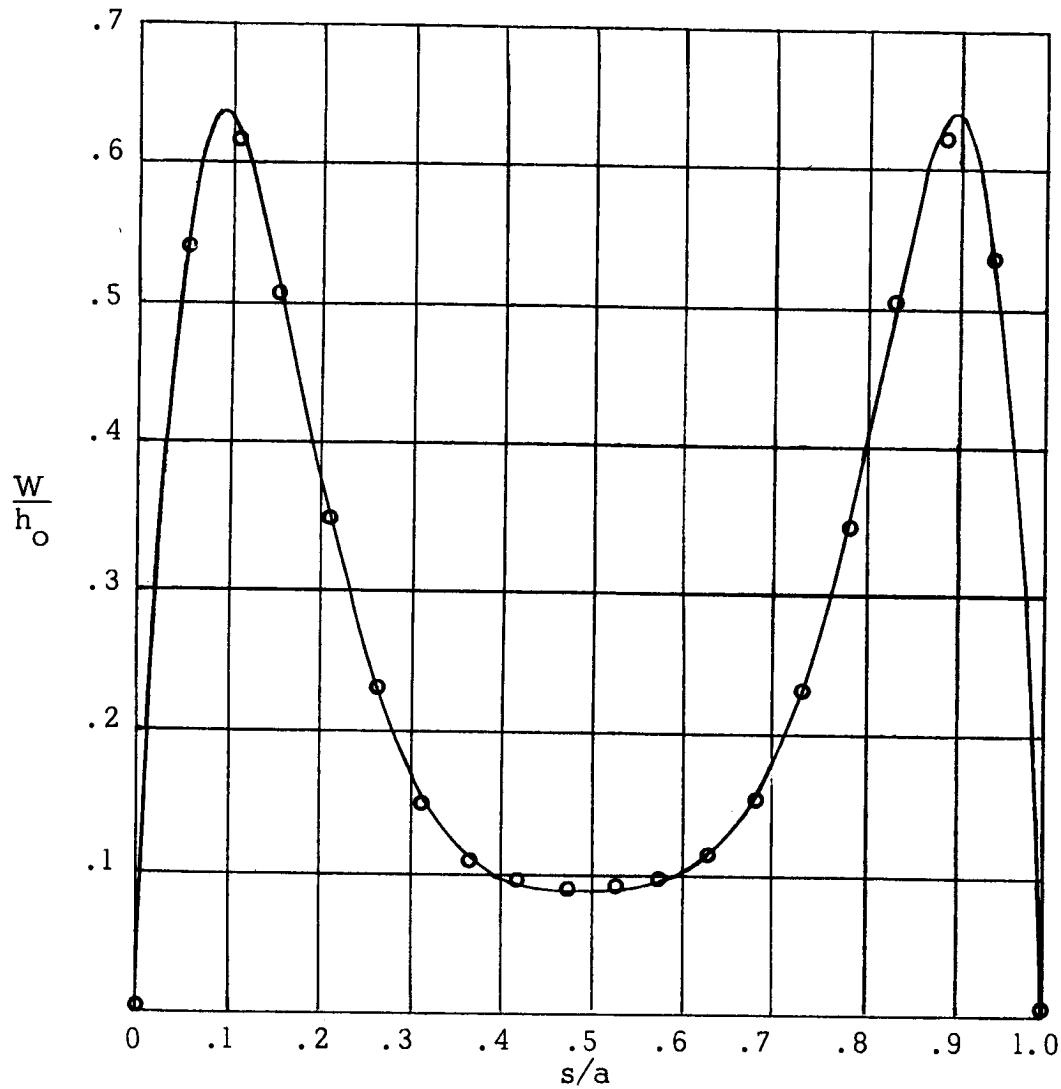


Figure 7. Deflection vs. Length for a Cylinder with $U=V=W=0$ and
 $(1-\nu^2)M_s/Eh_0^2 = 10^{-2} \cos 2\theta$ at Both Ends

— Reference 3, 300 points
 ○ This program, 20 points

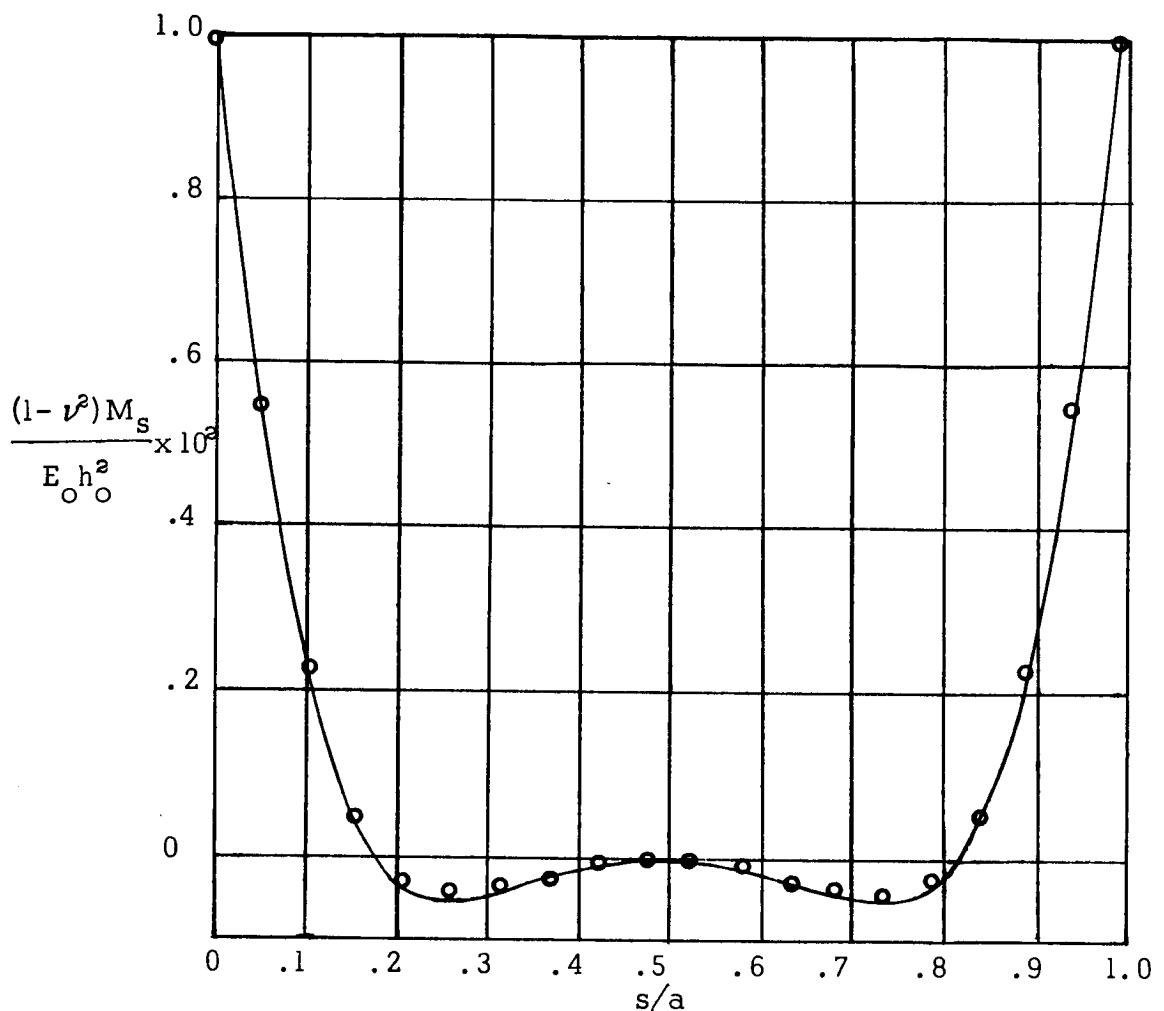


Figure 8. Moment vs. Length for a Cylinder with $U=V=W=0$ and
 $(1-\nu^2)M_s/Eh_0^2 = 10^{-2} \cos 2\theta$ at Both Ends

— Reference 3, 300 points

○ This program, 20 points

figure 9 where $P = q/q_{cr}$ and $q_{cr} = 4Eh^2/(R_s^2 \sqrt{12(1-\nu^2)})$, the classical buckling load of a uniformly loaded sphere. The quantity ($W_{1,1}$) is the normal displacement at station 1 on meridian 1. Station 1 is one-half of an increment length away from the apex, and it is assumed that meridian 1 is centered under the load. Similar results for the normal displacements at the apex of the shell and 1/3 the way down the meridian centered under the load from the analysis presented here are also given in figure 9. In order to provide a network with approximately the same degree of coarseness as that used in reference 1, a total of 4 circumferential modes, $n = 0, 1, 2$, and 3 were allowed with nonzero Fourier components of the applied load in the $n = 0, 1$ and 3 modes. Seven stations down the meridian were used. Figure 10 shows the normal displacement at each station along the $\theta = 0^\circ - 180^\circ$ meridian, and figure 11 presents, for each value of n , the normal displacement at the 1/3 point on the meridian centered under the load. The time of execution for this problem was less than .04 hours on an IBM 7094.

Famili and Archer (ref. 1) obtained a buckling load $P \approx .71$, while the buckling load from this program was $P \approx .66$. This is a good agreement considering the coarseness of the network and the fundamental differences in the theory and method of solution.

CONCLUSIONS

A digital computer program for predicting snap buckling of asymmetrically loaded shells of revolution has been written in the FORTRAN IV language for operation on the IBSYS-IBJOB operating system (version 13). An example problem of a clamped shallow spherical shell uniformly loaded over one-half of its surface was executed and the results were compared with a previous solution (ref. 1). Good agreement between the two solutions was observed. Consequently, the program appears to be operating correctly.

The execution time is not unduly excessive; a problem using the full capabilities of the program may take approximately .25 hours. However, the program is limited to 20 stations along the meridian and 10 circumferential modes. Consequently, only smooth shells can be adequately handled. The maximum number of stations and modes is determined by the size of core. Thus, when larger cores become available the program can easily be modified to increase the maximum number of stations and modes.

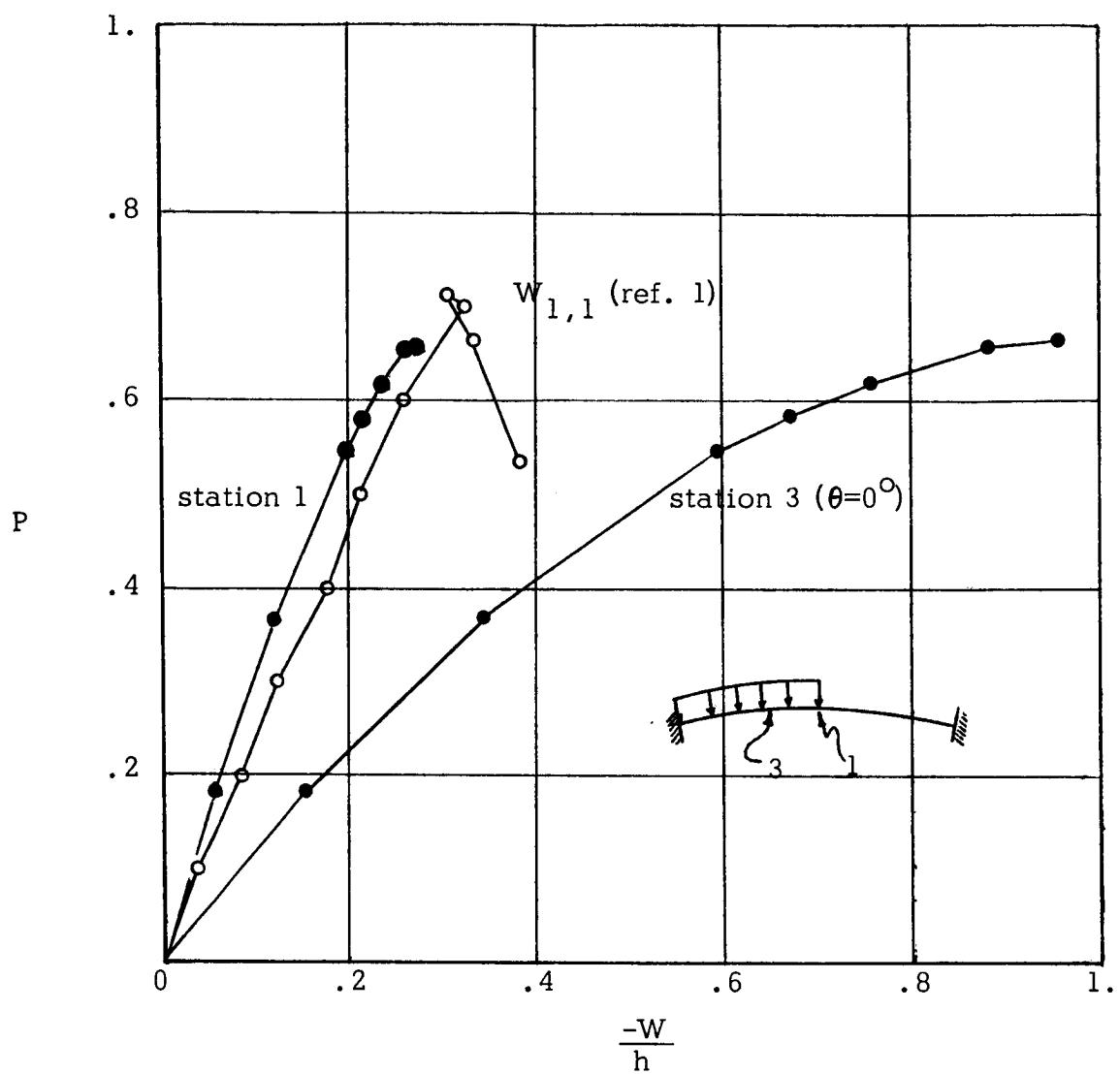


Figure 9. Load-deflection Curve for Half Loaded Shell

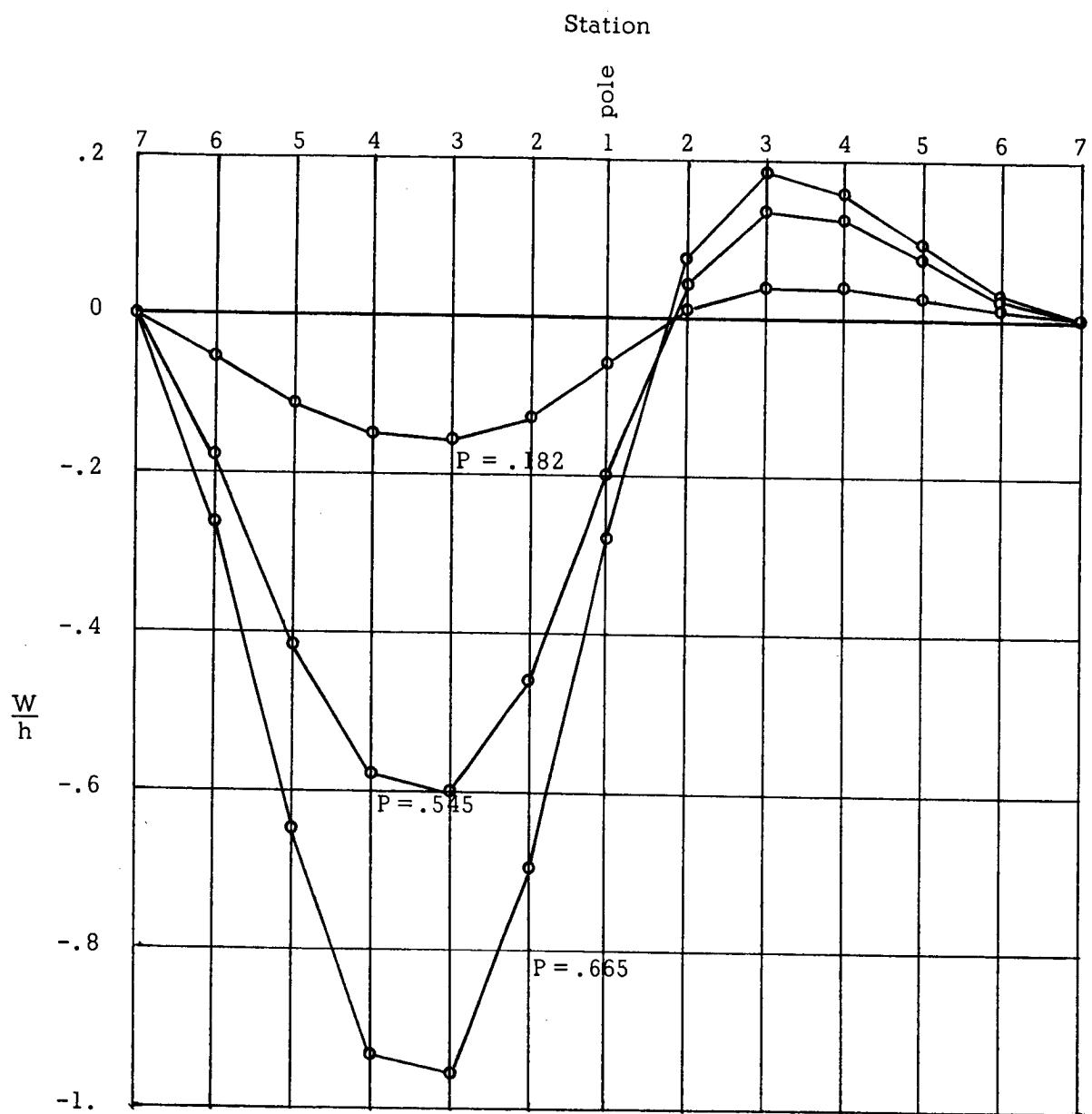


Figure 10. Deflection Profiles Along the $0 - 180^\circ$ Meridian for Half Loaded Shell.

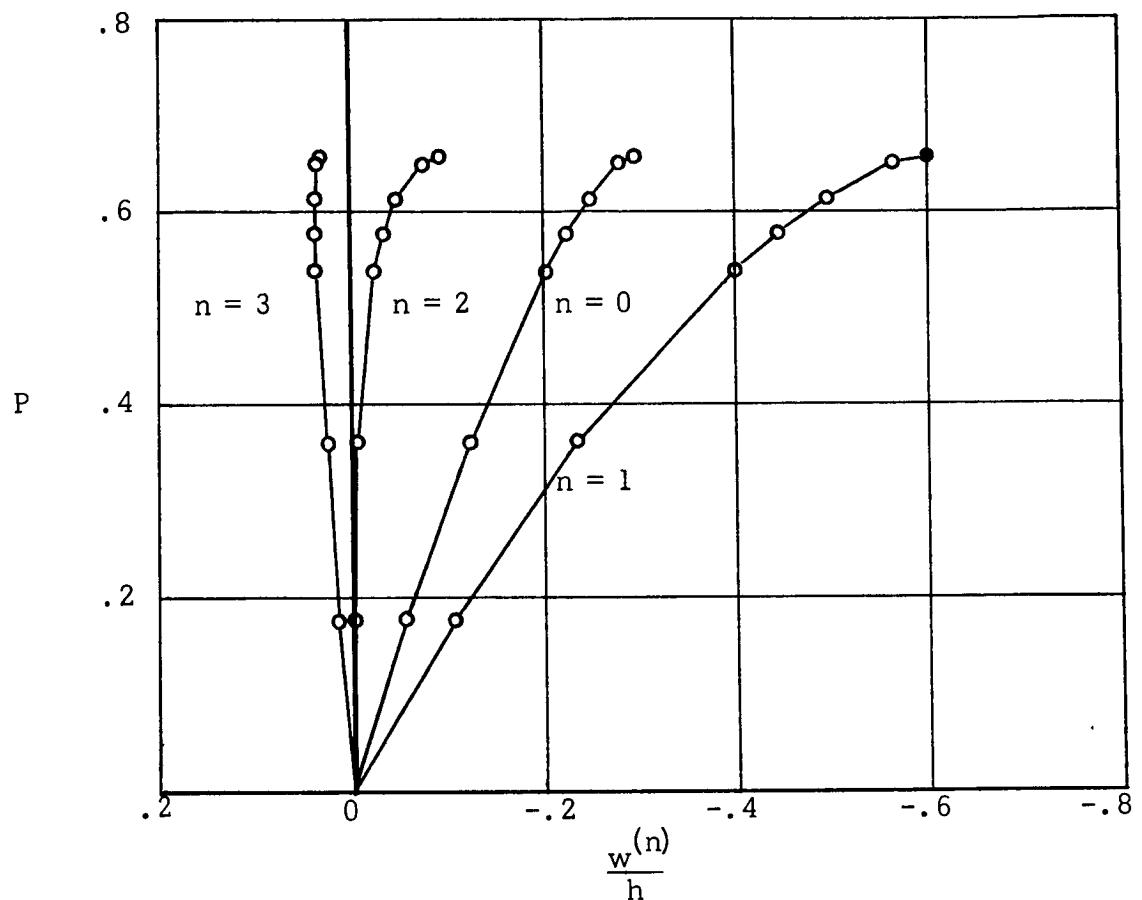


Figure 11. Normal Displacement for Each Value of n at Station 3 of Meridian Centered Under Load.

APPENDIX A

SANDERS' NONLINEAR EQUATIONS FOR THE GENERAL SHELL

Sanders (ref. 2) has derived a set of nonlinear equations for the general thin elastic shell based upon the assumption of small strains and moderately small rotations of the reference surface. His equations in conventional notation with lines of curvature for coordinates are reproduced here for the purpose of reference.

The strain-displacement relations are

$$\epsilon_{11} = (\alpha_1 \alpha_2)^{-1} \left[\alpha_2 u_{1,1} + \alpha_{1,2} u_2 + \alpha_1 \alpha_2 R_1^{-1} w + 1/2 \alpha_1 \alpha_2 \varphi_1^2 \right. \\ \left. + 1/2 \alpha_1 \alpha_2 \varphi^{2*} \right] \quad (A-1)$$

$$\epsilon_{12} = 1/2 (\alpha_1 \alpha_2)^{-1} \left[\alpha_2 u_{2,1} + \alpha_1 u_{1,2} - \alpha_{1,2} u_1 - \alpha_{2,1} u_2 \right. \\ \left. + \alpha_1 \alpha_2 \varphi_1 \varphi_2 \right] \quad (A-2)$$

and

$$\kappa_{11} = (\alpha_1 \alpha_2)^{-1} \left[\alpha_2 \varphi_{1,1} + \alpha_{1,2} \varphi_2 \right] \quad (A-3)$$

$$\kappa_{12} = 1/2 (\alpha_1 \alpha_2)^{-1} \left[\alpha_2 \varphi_{2,1} + \alpha_1 \varphi_{1,2} - \alpha_{1,2} \varphi_1 - \alpha_{2,1} \varphi_2 \right. \\ \left. + \alpha_1 \alpha_2 (R_2^{-1} - R_1^{-1}) \varphi^* \right] \quad (A-4)$$

where $\epsilon_{\gamma\beta}$ and $\kappa_{\gamma\beta}$ ($\gamma = 1, 2$, $\beta = 1, 2$) are the reference surface membrane and bending strains respectively, u_γ are the displacements tangent to the reference surface, w is the displacement normal to that surface, R_γ are the radii of curvature, α_γ are the Lamé coefficients, and a comma denotes partial differentiation with respect to the non-dimensional coordinates ξ_1 or ξ_2 as the subscript following

APPENDIX A

the comma indicates. The rotations φ_γ and φ are given by

$$\varphi_1 = -\alpha_1^{-1} w_{,1} + R_1^{-1} u_1^{**} \quad (A-5)$$

$$\varphi = 1/2(\alpha_1 \alpha_2)^{-1} [(\alpha_2 u_2)_{,1} - (\alpha_1 u_1)_{,2}] \quad (A-6)$$

Here, as in the following, the missing equations may be obtained by interchanging the subscripts 1 and 2 and changing the sign of φ .

The equations expressing equilibrium of forces and moments in directions parallel to the tangents and normal of the undeformed shell are

$$\begin{aligned} & (\alpha_2 N_{11})_{,1} + (\alpha_1 N_{12})_{,2} + \alpha_{1,2} N_{12} - \alpha_{2,1} N_{22} + \alpha_1 \alpha_2 R_1^{-1} Q_1^{**} \\ & + 1/2 \alpha_1 [(R_1^{-1} - R_2^{-1}) M_{12}]^*_{,2} - \alpha_1 \alpha_2 R_1^{-1} (\varphi_1 N_{11} \\ & + \varphi_2 N_{12})^{**} - 1/2 \alpha_1 [\varphi (N_{11} + N_{22})]^*_{,2} + \alpha_1 \alpha_2 p_1 = 0 \end{aligned} \quad (A-7)$$

$$\begin{aligned} & (\alpha_2 Q_1)_{,1} + (\alpha_1 Q_2)_{,2} - \alpha_1 \alpha_2 (R_1^{-1} N_{11} + R_2^{-1} N_{22}) - (\alpha_2 \varphi_1 N_{11} \\ & + \alpha_2 \varphi_2 N_{12})_{,1} - (\alpha_1 \varphi_1 N_{12} + \alpha_1 \varphi_2 N_{22})_{,2} + \alpha_1 \alpha_2 p = 0 \end{aligned} \quad (A-8)$$

$$(\alpha_2 M_{11})_{,1} + (\alpha_1 M_{12})_{,2} + \alpha_{1,2} M_{12} - \alpha_{2,1} M_{22} - \alpha_1 \alpha_2 Q_1 = 0 \quad (A-9)$$

where Q_γ are the transverse shear forces, p_γ are the tangential components of the applied pressure load, p is the normal component of the pressure load, and $N_{\gamma\beta}$ and $M_{\gamma\beta}$ are the membrane forces

APPENDIX A

and bending moments respectively. The conditions to be prescribed on an edge $\xi_1 = \text{constant}$ are

$$\begin{aligned}
 & N_{11} \quad \text{or} \quad u_1 \\
 & N_{12} + 1/2 \left(3 R_2^{-1} - R_1^{-1} \right) M_{12}^{**} + 1/2 \left(N_{11} + N_{22} \right) \varphi^* \quad \text{or} \quad u_2 \\
 & Q_1 + \alpha_2^{-1} M_{12,2} - \varphi_1 N_{11} - \varphi_2 N_{12} \quad \text{or} \quad w \\
 & M_{11} \quad \text{or} \quad \varphi_1
 \end{aligned} \tag{A-10}$$

When the rotations around the normal are neglected, the terms marked by the single asterisk drop out. The Donnell-Mushtari-Vlasov non-linear theory can be obtained by dropping out those terms marked with a single or a double asterisk.

APPENDIX B

NONLINEAR TERMS

Consider the strain-displacement relation for ϵ_s given by the first of equations (5). Applying the appropriate Fourier expansions to ϵ_s leads to

$$\sum_{n=0}^{\infty} \epsilon_s^{(n)} \cos n\theta = \sum_{n=0}^{\infty} (u^{(n)})' \cos n\theta + \omega_s \sum_{n=0}^{\infty} w^{(n)} \cos n\theta \\ + \frac{\sigma_o}{2E_o} \left\{ \left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right)^2 + \left(\sum_{n=1}^{\infty} \varphi_s^{(n)} \sin n\theta \right)^2 \right\} \quad (B-1)$$

The prime indicates differentiation with respect to the nondimensional coordinate $\xi = s/a$. The first two series on the right hand side of equation (B-1) can be immediately incorporated into the series on the left hand side, but the nonlinear terms must first be converted to single cosine series expansions before they can be included.

Consider the first nonlinear term in equation (B-1)

$$\left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right)^2 = \sum_{j=0}^{\infty} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(j)} \cos i\theta \cos j\theta \quad (B-2)$$

The product of the two cosines can be expressed as

$$\cos i\theta \cos j\theta = [\cos(i-j)\theta + \cos(i+j)\theta]/2$$

and so equation (B-2) becomes

$$\left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right)^2 = \left[\sum_{j=0}^{\infty} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(j)} \cos(i-j)\theta \right. \\ \left. + \sum_{j=0}^{\infty} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(j)} \cos(i+j)\theta \right] /2 \quad (B-3)$$

APPENDIX B

If the index j is replaced in the first double series with $i - n$, where $-\infty \leq n \leq i$, and with $n - i$, where $i \leq n \leq +\infty$, in the second double series, equation (B-3) can be given in the form

$$\begin{aligned} \left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right)^2 &= \left[\sum_{n=-\infty}^{-1} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(i-n)} \cos n\theta \right. \\ &\quad \left. + \sum_{n=0}^1 \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(i-n)} \cos n\theta + \sum_{n=1}^{\infty} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(n-i)} \cos n\theta \right] / 2 \end{aligned} \quad (B-4)$$

Combining the last two double series expansions in equation (B-4) into one double series, and noting that $\cos(-n\theta) = \cos n\theta$, leads to

$$\begin{aligned} \left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right)^2 &= \left[\sum_{n=1}^{\infty} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{(i+n)} \cos n\theta \right. \\ &\quad \left. + \sum_{n=0}^{\infty} \sum_{i=0}^{\infty} \varphi_s^{(i)} \varphi_s^{|i-n|} \mu^c \cos n\theta \right] / 2 \end{aligned} \quad (B-5)$$

where $\mu^c = 1$ for $i \neq n$ and $\mu^c = 2$ for $i = n$. Finally, combining both double series expansions in equation (B-5) to a single double series yields

$$\left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right)^2 = \frac{E_o}{\sigma_o} \sum_{n=0}^{\infty} \beta_s^{(n)} \cos n\theta \quad (B-6)$$

where

$$\beta_s^{(n)} = \frac{\sigma_o}{2E_o} \sum_{i=0}^{\infty} \varphi_s^{(i)} (\eta^c \varphi_s^{(i+n)} + \mu^c \varphi_s^{|i-n|}) \quad (B-7)$$

and $\eta^c = 0$ for $n = 0$ and $\eta^c = 1$ for $n > 0$.

It can be shown in a similar manner that the second nonlinear term in equation (B-1) reduces to

$$\left(\sum_{n=1}^{\infty} \varphi_s^{(n)} \sin n\theta \right)^2 = \frac{E_o}{\sigma_o} \sum_{n=0}^{\infty} \beta_s^{(n)} \cos n\theta \quad (B-8)$$

APPENDIX B

where

$$\beta^{(n)} = \frac{\sigma_0}{2E_0} \sum_{i=1}^{\infty} \varphi^{(i)} (\eta^s \varphi^{(i+n)} + \mu^s \varphi^{(i-n)}) \quad (B-9)$$

and

$$\mu^s = 1 \text{ when } i \geq n+1 \quad \eta^s = 0 \text{ when } n=0$$

$$\mu^s = 0 \text{ when } i=n \quad \eta^s = 1 \text{ when } n>0$$

$$\mu^s = -1 \text{ when } i \leq n-1$$

Therefore, equation (B-1) can now be put into the form

$$\sum_{n=0}^{\infty} \left[e_s^{(n)} - (u^{(n)})' - \omega_s w^{(n)} - (\beta_s^{(n)} + \beta^{(n)})/2 \right] \cos n\theta = 0$$

Hence,

$$e_s^{(n)} = (u^{(n)})' + \omega_s w^{(n)} + (\beta_s^{(n)} + \beta^{(n)})/2 \quad (B-10)$$

Finally, the two remaining nonlinear terms in the strain-displacement equations Φ_θ^2 and $\Phi_s \Phi_\theta$ reduce to

$$\Phi_\theta^2 = \left(\frac{\sigma_0}{E_0} \right)^2 \left(\sum_{n=1}^{\infty} \varphi_\theta^{(n)} \sin n\theta \right)^2 = \left(\frac{\sigma_0}{E_0} \right) \sum_{n=0}^{\infty} \beta_\theta^{(n)} \cos n\theta \quad (B-11)$$

$$\begin{aligned} \Phi_s \Phi_\theta &= \left(\frac{\sigma_0}{E_0} \right)^2 \left(\sum_{n=0}^{\infty} \varphi_s^{(n)} \cos n\theta \right) \left(\sum_{n=1}^{\infty} \varphi_\theta^{(n)} \sin n\theta \right) \\ &= \left(\frac{\sigma_0}{E_0} \right) \sum_{n=1}^{\infty} \beta_{s\theta}^{(n)} \sin n\theta \end{aligned} \quad (B-12)$$

where $\beta_\theta^{(n)}$ is of the same form as $\beta^{(n)}$ given by equation (B-9) and

APPENDIX B .

$$\beta_{s\theta}^{(n)} = \frac{\sigma_o}{2E_o} \sum_{i=0}^{\infty} \varphi_s^{(i)} (\varphi_\theta^{(i+n)} + \mu^{sc} \varphi_\theta^{|i-n|}) \quad (B-13)$$

with

$$\mu^{sc} = -1 \text{ when } i \geq n + 1$$

$$\mu^{sc} = 0 \text{ when } i = n$$

$$\mu^{sc} = 1 \text{ when } i \leq n - 1$$

For η_{ss} , equation (B-7) applies when $\varphi_s^{(i)}$ is replaced with $t_s^{(i)}$. Similarly, equation (B-9) applies for $\eta_{\theta s}$ and equation (B-13) applies for η_s , η_θ , $\eta_{\theta\theta}$, and $\eta_{s\theta}$ when the appropriate rotation is replaced with the appropriate force.

For a given value of n , equations (B-7, B-9, and B-13) define all the sets of Fourier indices that combine through the product of two series to give n . Thus, when the φ 's and t 's are known for all values of n , the β 's and η 's can be calculated for each n using these three equations. However, because of the possibility that the φ 's and t 's will be zero for many values of i , a more efficient procedure for calculating the β 's and η 's is actually used. This procedure is carried out in the computer program subroutines MODES, PHIBET and TEAETA. For a discussion and flow chart of these subroutines see Appendix E, and Appendix G.

APPENDIX C

MATRIX EXPRESSIONS

The expressions for the elements in the E, F and G matrices are:

$$E_{11} = b \quad (C-1)$$

$$E_{22} = \frac{b(1-\nu)}{2} + \frac{\lambda^2 d(1-\nu)(3\omega_\theta - \omega_s)^2}{8} \quad (C-2)$$

$$E_{23} = \frac{\lambda^2 d(1-\nu)(3\omega_\theta - \omega_s) n}{2\rho} \quad (C-3)$$

$$E_{32} = E_{23} \quad (C-4)$$

$$E_{33} = \lambda^2 d(1-\nu) \left[\left(2n^2/\rho^2 \right) + (1+\nu) \gamma^2 \right] \quad (C-5)$$

$$E_{34} = \lambda^2 \quad (C-6)$$

$$E_{43} = -d \quad (C-7)$$

$$F_{11} = \gamma b + b' \quad (C-8)$$

$$F_{12} = \frac{(1+\nu)bn}{2\rho} + \frac{\lambda^2 dn(1-\nu)}{8\rho} (3\omega_s - \omega_\theta) (3\omega_\theta - \omega_s) \quad (C-9)$$

$$F_{13} = b(\omega_s + \nu \omega_\theta) + \lambda^2 d(1-\nu) \left[(1+\nu) \gamma^2 \omega_s \right]$$

APPENDIX C

$$+ \left(n^2 / 2\rho^2 \right) (3\omega_s - \omega_\theta) \Big] \quad (C-10)$$

$$F_{14} = \lambda^2 \omega_s \quad (C-11)$$

$$F_{21} = - F_{12} \quad (C-12)$$

$$\begin{aligned} F_{22} &= \left(\frac{1-\nu}{2} \right) (\gamma b + b') - \frac{\lambda^2 d(1-\nu)}{8} (3\omega_\theta - \omega_s) \left[2\omega_s' \right. \\ &\quad \left. - \gamma (5\omega_s - 3\omega_\theta) \right] + \frac{\lambda^2 d'(1-\nu)}{8} (3\omega_\theta - \omega_s)^2 \end{aligned} \quad (C-13)$$

$$\begin{aligned} F_{23} &= \frac{\lambda^2 d(1-\nu)n}{2\rho} \left[2(1+\nu)\gamma\omega_\theta - \omega_s' + 3\gamma(\omega_s - \omega_\theta) \right] \\ &\quad + \frac{\lambda^2 d'(1-\nu)(3\omega_\theta - \omega_s)n}{2\rho} \end{aligned} \quad (C-14)$$

$$F_{31} = - F_{13} \quad (C-15)$$

$$\begin{aligned} F_{32} &= \frac{\lambda^2 d(1-\nu)n}{2\rho} \left[3\gamma\omega_s - \gamma\omega_\theta(5+2\nu) - \omega_s' \right] \\ &\quad + \frac{\lambda^2 d'(1-\nu)n}{2\rho} (3\omega_\theta - \omega_s) \end{aligned} \quad (C-16)$$

$$\begin{aligned} F_{33} &= -\lambda^2 d(1-\nu) \left[(1+\nu) \left(2\gamma\omega_s\omega_\theta + \gamma^3 \right) \right. \\ &\quad \left. + \left(2\gamma n^2 / \rho^2 \right) \right] + \lambda^2 d'(1-\nu) \left[(1+\nu)\gamma^2 + \left(2n^2 / \rho^2 \right) \right] \end{aligned} \quad (C-17)$$

APPENDIX C

$$F_{34} = \lambda^2 \gamma (2 - \nu) \quad (C-18)$$

$$F_{41} = d \omega_s \quad (C-19)$$

$$F_{43} = -d \nu \gamma \quad (C-20)$$

$$G_{11} = \nu b' \gamma - \nu b \omega_s \omega_\theta - b \gamma^2 - \frac{(1 - \nu)b n^2}{2 \rho^2}$$

$$- \lambda^2 d(1 - \nu) \left[(1 + \nu) \gamma^2 \omega_s^2 + \frac{(3\omega_s - \omega_\theta)^2 n^2}{8 \rho^2} \right] \quad (C-21)$$

$$G_{12} = \frac{\nu n b'}{\rho} - \left(\frac{3 - \nu}{2 \rho} \right) (\gamma b n) - \frac{\lambda^2 d(1 - \nu) \gamma n}{\rho} \left[\frac{(3\omega_s - \omega_\theta)(3\omega_\theta - \omega_s)}{8} \right.$$

$$\left. + (1 + \nu) \omega_s \omega_\theta \right] \quad (C-22)$$

$$G_{13} = b \left[\omega_s' + \gamma (\omega_s - \omega_\theta) \right] + b' (\omega_s + \nu \omega_\theta)$$

$$- \frac{\lambda^2 d(1 - \nu) \gamma n^2}{\rho^2} \left[\frac{3\omega_s - \omega_\theta}{2} + (1 + \nu) \omega_s \right] \quad (C-23)$$

$$G_{14} = \lambda^2 (1 - \nu) \gamma \omega_s \quad (C-24)$$

$$G_{21} = -\frac{b \gamma n}{2 \rho} (3 - \nu) - \frac{(1 - \nu) n b'}{2 \rho} + \frac{\lambda^2 d(1 - \nu) n}{\rho} \left[-(1 + \nu) \gamma \omega_s \omega_\theta \right.$$

$$\left. + \frac{\gamma}{8} (6 \omega_s \omega_\theta - 7 \omega_s^2 - 3 \omega_\theta^2) - \frac{\omega_s'}{4} (5 \omega_\theta - 3 \omega_s) \right]$$

$$- \frac{\lambda^2 d' (1 - \nu) n}{8 \rho} (3 \omega_s - \omega_\theta)(3 \omega_\theta - \omega_s) \quad (C-25)$$

APPENDIX C

$$G_{22} = -\gamma F_{22}$$

$$+ \left(\frac{1-\nu}{2} \right) b \omega_s \omega_\theta - \frac{bn^2}{\rho^2} - \lambda^2 d (1-\nu) \left[\frac{(1+\nu) \omega_\theta^2 n^2}{\rho^2} \right.$$

$$\left. - \frac{\omega_s \omega_\theta}{8} (3\omega_\theta - \omega_s)^2 \right] \quad (C-26)$$

$$G_{23} = \frac{-bn(\omega_\theta + \nu \omega_s)}{\rho} + \frac{\lambda^2 dn(1-\nu)}{2\rho} \left[\gamma \omega_s' - 2\gamma^2 \omega_s \right.$$

$$\left. - \frac{2(1+\nu) \omega_\theta n^2}{\rho^2} + (3\omega_\theta - \omega_s) (\gamma^2 + \omega_s \omega_\theta) \right]$$

$$- \frac{\lambda^2 d'n(1-\nu) \gamma}{2\rho} (3\omega_\theta - \omega_s) \quad (C-27)$$

$$G_{24} = -(\nu \lambda^2 \omega_\theta n / \rho)$$

(C-28)

$$G_{31} = -b \gamma (\omega_\theta + \nu \omega_s) + \lambda^2 d (1-\nu) \left[\gamma (1+\nu) (-\gamma \omega_s' \right.$$

$$+ \gamma^2 \omega_s - (n^2 \omega_s / \rho^2) + 2 \omega_s^2 \omega_\theta \left. \right) + (n^2 / 2 \rho^2) (\gamma \omega_s$$

$$- \gamma \omega_\theta - 3 \omega_s') \left. \right] - \lambda^2 d' (1-\nu) \left[(1+\nu) \gamma^2 \omega_s \right.$$

$$+ (n^2 / 2 \rho^2) (3 \omega_s - \omega_\theta) \left. \right] \quad (C-29)$$

$$G_{32} = - \frac{bn(\omega_\theta + \nu \omega_s)}{\rho} + \frac{\lambda^2 d(1-\nu)n}{2\rho} \left[2(1+\nu) (\omega_s \omega_\theta^2 \right.$$

APPENDIX C

$$\begin{aligned}
 & -\gamma^2 \omega_s + 2\gamma^2 \omega_\theta - \frac{n^2 \omega_\theta}{\rho^2} \Big) + \gamma \omega_s' + 3\gamma^2 (\omega_\theta - \omega_s) \\
 & + \omega_s \omega_\theta (3\omega_\theta - \omega_s) \Big] - \frac{\lambda^2 d(1-\nu)n}{2\rho} \left[2(1+\nu)\gamma \omega_\theta \right. \\
 & \left. + \gamma (3\omega_\theta - \omega_s) \right] \quad (C-30)
 \end{aligned}$$

$$\begin{aligned}
 G_{33} = & -b(\omega_s^2 + 2\nu \omega_s \omega_\theta + \omega_\theta^2) + \frac{\lambda^2 d(1-\nu)n^2}{\rho^2} \left[(1+\nu)(\omega_s \omega_\theta \right. \\
 & \left. - \frac{n^2}{\rho^2} + 2\gamma^2) + 2(\gamma^2 + \omega_s \omega_\theta) \right] - \frac{\lambda^2 d(1-\nu)n^2}{\rho^2} (3+\nu)\gamma \quad (C-31)
 \end{aligned}$$

$$G_{34} = -\lambda^2 \left[(1-\nu) \omega_s \omega_\theta + \left(\nu n^2 / \rho^2 \right) \right] \quad (C-32)$$

$$G_{41} = d \left(\omega_s' + \nu \gamma \omega_s \right) \quad (C-33)$$

$$G_{42} = d \nu n \omega_\theta / \rho \quad (C-34)$$

$$G_{43} = d \nu n^2 / \rho^2 \quad (C-35)$$

$$G_{44} = -1 \quad (C-36)$$

All other elements are zero.

The expressions for the elements in the e column matrix are:

$$e_1 = -p_s + t'_T - \lambda^2 (1-\nu) \gamma \omega_s m_T - \left\{ b (\beta_s' + \beta') \right.$$

APPENDIX C

$$\begin{aligned}
 & + \left[b' + \gamma b(1 - \nu) \right] (\beta_s + \beta) + \nu b (\beta_{\theta}' + \beta') \\
 & + \left[\nu b' - \gamma b(1 - \nu) \right] (\beta_{\theta} + \beta) + (n/\rho) b (1 - \nu) \beta_{s\theta} \\
 & - 2\omega_s (\eta_{ss} + \eta_{\theta s}) - n (\eta_s + \eta_{\theta})/\rho \} / 2 \tag{C-37}
 \end{aligned}$$

$$\begin{aligned}
 e_2 = & - p_{\theta} - (n/\rho) t_T - \lambda^2 (1 - \nu) (n/\rho) \omega_{\theta} m_T + \left\{ (n/\rho) b (\beta_{\theta} + \beta) \right. \\
 & + \nu b (n/\rho) (\beta_s + \beta) - (1 - \nu) b \beta_{s\theta}' - (1 - \nu) (b' + 2\gamma b) \beta_{s\theta} \\
 & \left. + 2\omega_{\theta} (\eta_{\theta\theta} + \eta_{s\theta}) - (\eta_s' + \eta_{\theta}') \right\} / 2 \tag{G-38}
 \end{aligned}$$

$$\begin{aligned}
 e_3 = & - p - (\omega_s + \omega_{\theta}) t_T - \lambda^2 (1 - \nu) \gamma m_T' + \lambda^2 (1 - \nu) \left[\omega_s \omega_{\theta} \right. \\
 & - (n/\rho)^2 \left. \right] m_T + \left\{ b \left[(\omega_s + \nu \omega_{\theta}) (\beta_s + \beta) \right. \right. \\
 & + (\omega_{\theta} + \nu \omega_s) (\beta_{\theta} + \beta) \left. \right] + 2 \left[\gamma (\eta_{ss} + \eta_{\theta s}) \right. \\
 & \left. + (\eta_{ss} + \eta_{\theta s})' \right] + 2(n/\rho) (\eta_{s\theta} + \eta_{\theta\theta}) \right\} / 2 \tag{C-39}
 \end{aligned}$$

$$e_4 = m_T \tag{C-40}$$

APPENDIX C

The expressions for the elements of the H and J matrices are:

$$H_{11} = b \quad (C-41)$$

$$H_{22} = \frac{b(1-\nu)}{2} + \frac{\lambda^2 d(1-\nu)}{8} (3\omega_\theta - \omega_s)^2 \quad (C-42)$$

$$H_{23} = \frac{\lambda^2 d(1-\nu) n}{2\rho} (3\omega_\theta - \omega_s) \quad (C-43)$$

$$H_{32} = H_{23} \quad (C-44)$$

$$H_{33} = \lambda^2 d(1-\nu) \left[2 \left(\frac{n}{\rho} \right)^2 + (1+\nu) \gamma^2 \right] \quad (C-45)$$

$$H_{34} = \lambda^2 \quad (C-46)$$

$$H_{43} = -1 \quad (C-47)$$

$$J_{11} = \nu \gamma b \quad (C-48)$$

$$J_{12} = \frac{\nu n b}{\rho} \quad (C-49)$$

$$J_{13} = b (\omega_s + \nu \omega_\theta) \quad (C-50)$$

$$J_{21} = -\frac{b(1-\nu) n}{2\rho} - d \frac{\lambda^2 (1-\nu)}{8\rho} n (3\omega_s - \omega_\theta) (3\omega_\theta - \omega_s) \quad (C-51)$$

$$J_{22} = -\gamma H_{22} \quad (C-52)$$

$$J_{23} = -\gamma H_{23} \quad (C-53)$$

APPENDIX C

$$J_{31} = -\lambda^2 d (1-\nu) \left[(1+\nu) \gamma^2 \omega_s + \frac{n^2}{2\rho^2} (3\omega_s - \omega_\theta) \right] \quad (C-54)$$

$$J_{32} = -\frac{\lambda^2 d (1-\nu)}{2\rho} \gamma n \left[3\omega_\theta - \omega_s + 2(1+\nu)\omega_\theta \right] \quad (C-55)$$

$$J_{33} = -\lambda^2 d (1-\nu) (3+\nu) \frac{\gamma n^2}{\rho^2} \quad (C-56)$$

$$J_{34} = \lambda^2 (1-\nu) \gamma \quad (C-57)$$

$$J_{41} = \omega_s \quad (C-58)$$

All other elements are zero.

The expressions for the elements in the f column matrix are:

$$f_1 = -t_T + b \left[\beta_s + \nu \beta_\theta + (1+\nu) \beta \right] / 2 \quad (C-59)$$

$$f_2 = \left[b (1-\nu) \beta_{s\theta} + \eta_s + \eta_\theta \right] / 2 \quad (C-60)$$

$$f_3 = \lambda^2 \gamma (1-\nu) m_T - (\eta_{ss} + \eta_{\theta s}) \quad (C-61)$$

$$f_4 = 0 \quad (C-62)$$

APPENDIX D

POLES

At a pole $\rho = 0$ when $i = 1$ or $i = K$ and consequently special conditions on u , v , w and m_s must be imposed to assure finite strains and equilibrium. Several sets of pole conditions have been proposed for the linear shell theory in reference 3. In this section the conditions associated with the nonlinear theory are derived, and the associated matrices are formulated. For an initial pole

$$z_1 = -P_1 z_2 + x_1$$

where P_1 and x_1 are determined from the pole conditions. For a final pole

$$z_{K-1} = -P_{K-1} z_K + x_{K-1} \quad (D-1)$$

$$B_K z_K + C_K z_{K-1} = g_K \quad (D-2)$$

where P_{K-1} and x_{K-1} are evaluated using equations (49a and 49b) and B_K , C_K , and g_K are determined from the pole conditions.

Solving equations (D-1 and D-2) for z_K gives

$$z_K = [B_K - C_K P_{K-1}]^{-1} \{g_K - C_K x_{K-1}\} \quad (D-3)$$

and z_{K-1} , z_{K-2} , ... can be found from equation (47).

Assuming a smooth reference surface over the pole

$$\rho' = 1 \quad (D-4)$$

$$\rho'' = 0 \quad (D-5)$$

$$\omega_\theta = \omega_s \quad (D-6)$$

when $\rho = 0$. Substituting equations (D-4, D-5, and D-6) and $\rho = 0$ into equations (27, 28, and 29) and applying L'Hopital's rule result in

APPENDIX D

$$e_s = u' + \omega_s w + (\beta_s + \beta)/2 \quad (D-7)$$

$$e_\theta = n v' + u' + \omega_\theta w + (\beta_\theta + \beta)/2 \quad (D-8)$$

$$e_{s\theta} = (-nu' + \beta_{s\theta})/2 \quad (D-9)$$

$$k_s = \varphi'_s \quad (D-10)$$

$$k_\theta = n \varphi'_\theta + \varphi'_s \quad (D-11)$$

$$k_{s\theta} = -n \varphi'_s / 2 \quad (D-12)$$

$$\varphi_s = -w' + \omega_s u \quad (D-13)$$

$$\varphi_\theta = n w' + \omega_\theta v \quad (D-14)$$

$$\varphi = (2v' + nu')/2 \quad (D-15)$$

provided that

$$nv + u = nu + v = nw = 0 \quad (D-16)$$

$$n \varphi_\theta + \varphi_s = n \varphi_s + \varphi_\theta = 0 \quad (D-17)$$

Equations (D-16 and D-17) assure finite values for the strains, curvatures and slopes at the pole. According to equations (D-16 and D-17)

$$u = v = \varphi_\theta = \varphi_s = 0 \quad \text{for } n = 0 \quad (D-18)$$

$$u + v = \varphi_\theta + \varphi_s = w = 0 \quad \text{for } n = 1 \quad (D-19)$$

$$u = v = \varphi_\theta = \varphi_s = w = 0 \quad \text{for } n > 1 \quad (D-20)$$

Applying equation (D-17) to equations (D-13 and D-14) and taking account of equation (D-16) gives the additional condition

$$w' = 0 \quad \text{for } n \neq 1 \quad (D-21)$$

APPENDIX D

so that the slopes simplify to

$$\left. \begin{aligned} \varphi_s &= -w' + \omega_s u \\ \varphi_\theta &= w' + \omega_s v = -\varphi_s \\ \varphi &= (2v' + u')/2 \end{aligned} \right\} \text{for } n = 1$$

and

$$\left. \begin{aligned} \varphi_s &= \varphi_\theta = 0 \\ \varphi &= (2v' + n u')/2 \end{aligned} \right\} \text{for } n \neq 1$$

Consequently, it can easily be shown using results from Appendix B that

$$\beta_s = \beta_\theta = \beta_{s\theta} = 0 \quad \text{for } n \neq 0, 2 \quad (\text{D-22})$$

and

$$\beta_s = (-w' + \omega_s u)^2/2 \quad \text{for } n = 0, 2 \quad (\text{D-23})$$

$$\beta_\theta = (w' + \omega_s v)^2/2 \quad \text{for } n = 0 \quad (\text{D-24})$$

$$\beta_{s\theta} = - (w' + \omega_s v)^2/2 \quad \text{for } n = 2 \quad (\text{D-25})$$

$$\beta_{s\theta} = (-w' + \omega_s u)(w' + \omega_s v)/2 \quad \text{for } n = 2 \quad (\text{D-26})$$

where w' , u and v are the solution corresponding to $n = 1$.

Additional conditions to impose at the pole arise from consideration of equilibrium. Substituting the appropriate Fourier expansions into equations (9a and 9b) and applying the geometric conditions at a pole lead to

$$m_s + n m_{s\theta} - m_\theta = 0 \quad (\text{D-27})$$

$$-n m_\theta + 2 m_{s\theta} = 0 \quad (\text{D-28})$$

APPENDIX D

Substituting equations (33a, 33b, 33c, and D-10, D-11, D-12) into equations (D-27 and D-28) and simplifying gives*

$$n(2\varphi'_\theta + n\varphi'_s) = 0 \quad (D-29)$$

$$n(2\varphi'_s + n\varphi'_\theta) = 0 \quad (D-30)$$

Solving equations (D-29 and D-30) simultaneously give the conditions

$$\varphi'_s = \varphi'_\theta = 0 \quad \text{for } n \neq 0, 2 \quad (D-31)$$

$$\varphi'_s = -\varphi'_\theta \quad \text{for } n = 2 \quad (D-32)$$

and consequently,

$$m_s = m_\theta = 0 \quad \text{for } n \neq 0, 2 \quad (D-33)$$

Further, from equations (29)

$$\varphi'_s = -w'' + \omega_s u' + \omega'_s u \quad (D-34)$$

$$\varphi'_\theta = n(r w' - r' w)/r^2 + \omega_\theta v' + \omega'_\theta v \quad (D-35)$$

When $r = 0$, equation (D-35) reduces to

$$\varphi'_\theta = n w''/2 + \omega_s v' + \omega'_\theta v \quad (D-36)$$

after repeated application of L'Hopital's rule. Eliminating w'' from equations (D-34 and D-36) gives

$$\varphi'_\theta + n\varphi'_s/2 = \omega_s(v' + n u'/2) + \omega'_\theta v + n \omega'_s u/2 \quad (D-37)$$

However, from the Codazzi relationship, equation (3),

$$\omega'_\theta = \omega'_s/2$$

*The assumption is made that $m_T = 0$ at the pole for $n > 0$.

APPENDIX D

at a pole. Consequently, equation (D-37) simplifies to

$$\varphi'_\theta + n \varphi'_s / 2 = \omega_s (v' + n u'/2)$$

since $v + n u = 0$ for all n . Applying equation (D-31) gives

$$v' + n u'/2 = 0 \quad \text{for } n \neq 0, 2$$

so that

$$\varphi = 0 \quad \text{for } n \neq 2 \quad (\text{D-38})$$

according to equation (D-15). Additionally, substituting equations (D-34 and D-36) into equations (D-31 and D-32) results in

$$v' + u' = \varphi = 0 \quad \text{for } n = 2 \quad (\text{D-39})$$

since $u = v = 0$ when $n = 2$. Therefore, the nonlinear terms

$$\beta = \eta_s = \eta_\theta = 0 \quad \text{for } n \geq 0 \quad (\text{D-40})$$

and consequently, the first two of equations (30) reduce to

$$t_s + n t_{s\theta} - t_\theta = 0 \quad (\text{D-41})$$

$$-n t_\theta + 2 t_{s\theta} = 0 \quad (\text{D-42})$$

at the pole. Substituting equations (32 and D-7, D-8, D-9) with $n = 1$ into equations (D-41 and D-42) leads to*

$$u' = v' = 0 \quad \text{for } n = 1$$

Finally, on the basis of symmetry

$$m'_s = 0 \quad \text{for } n = 0, 2$$

Sufficient conditions have now been developed to determine the matrices P_1 and x_1 . Summarizing the results:

1) $m'_s = 0$ for $n = 0$

$$u = v = w' = m'_s = 0$$

*The assumption is made that $t_T = 0$ at the pole for $n > 0$.

APPENDIX D

and

$$t_s = b(1 + \nu) \left(u' + \omega_s w + \beta_s / 2 \right) - t_T$$

$$t_\theta = t_s$$

$$t_{s\theta} = 0$$

$$m_\theta = m_s$$

$$m_{s\theta} = 0$$

2) for $n = 1$

$$u + v = u' = w = m_s = 0$$

and

$$t_s = t_\theta = t_{s\theta} = m_\theta = m_{s\theta} = 0$$

3) for $n = 2$

$$u = v = w = m_s' = 0$$

and

$$t_s = b(1 - \nu) \left(u' + \beta_s / 2 \right)$$

$$t_\theta = t_{s\theta} = -t_s$$

$$m_\theta = m_{s\theta} = -m_s$$

4) for $n > 2$

$$u = v = w = m_s = 0$$

APPENDIX D

and

$$t_s = t_\theta = t_{s\theta} = m_\theta = m_{s\theta} = 0$$

where β_s is given by equation (D-23).

Based on these pole conditions and the approximation for the first derivative at an initial pole*

$$z'_1 = (z_2 - z_1) / \Delta$$

it can be stated that

$$x_1 = 0 \quad \text{for } n \geq 0$$

and

$$\begin{aligned} P_1 &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, & \text{for } n = 0 \\ P_1 &= \begin{bmatrix} -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} & \text{for } n = 1 \\ P_1 &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, & \text{for } n = 2 \\ P_1 &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} & \text{for } n > 2 \end{aligned}$$

At a final pole the approximation*

$$z'_K = (z_K - z_{K-1}) / \Delta$$

is assumed so that

$$g_K = 0 \quad \text{for } n \geq 0$$

and that for $n = 0$

$$\begin{aligned} B_K &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ C_K &= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \end{aligned}$$

*Note that this finite difference approximation to the first derivative at a pole has an error $O(\Delta)$ whereas the finite difference approximations to the first and second differential equations (36) have an error $O(\Delta^2)$. The effect of this inconsistency upon the solution is small.

APPENDIX D

for $n = 1$,

$$B_K = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad C_K = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

for $n = 2$,

$$B_K = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad C_K = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

and for $n > 2$

$$z_K = 0$$

APPENDIX E

PROGRAM WRITE-UP

The computer program has been written in FORTRAN IV language for operation in the IBSYS-IBJOB operating system (version 13). The physical model is limited to twenty equally spaced meridian stations and ten arbitrary Fourier terms. The alternate input-output feature of the IBJOB monitor (ALTIO) is required in order to conserve core space. When completely loaded the program fills all but four hundred words of the thirty-two thousand word memory of the IBM 7094. No programming is necessary if the user reads in data for the shell geometry, the inplane and bending stiffnesses, and the loading at each meridian station. However, the program does provide the user the option of programming an analytical model with minimum effort, if so desired.

During execution the program solves a set of nondimensional linearized equations for each Fourier coefficient of the actual load, plus an estimated pseudo load from the nonlinear terms, using a finite difference formulation with a Gaussian elimination procedure. The nonlinear terms are recalculated using the new solution and entered into the equations as revised pseudo loads. All sets of linearized equations are solved again. This procedure is repeated until convergence is achieved. A load-deformation history is automatically determined using a variable incremental load step to assure reasonable convergence of the iteration procedure as the load approaches the snap through value. Although the program operates internally on nondimensional quantities the input and output data are in dimensional form.

The problem of a spherical cap of constant thickness, fixed at its outer edge, and uniformly loaded over half its surface is used as a test case to illustrate the input and output features of the program*. For this problem $\nu = .3$, $R_s = R_\theta = 1000$ in, the total meridian length $S = 105$ in, $E_o = 30 \times 10^6$ lb/in², $\sigma_o = 1000$ lb/in², $B = 27.3 \times 10^6$ lb/in, $D = 2.275 \times 10^6$ lb-in, and $q = -30$ lb/in² over the segment $\theta = -90^\circ$ to $\theta = 90^\circ$ so that the first three Fourier terms of the load are $\frac{\sigma_o h}{a} p(0) = -15.0$ lb/in², $\frac{\sigma_o h}{a} p(1) = -19.1$ lb/in² and $\frac{\sigma_o h}{a} p(3) = 6.37$ lb/in². All other loads are zero. The boundary conditions at the outer edge are $U = V = W = \Phi_s = 0$. The classical buckling pressure of a uniformly loaded complete sphere with these same properties is 33.1 lb/in².

*See the section EXAMPLES for a more complete discussion of this problem.

APPENDIX E

Input Requirements

The card layout shown in figure 12 indicates the input data format for all cards. Cards numbered 1 - 48 are fixed input cards, i.e., these cards are required with each case.

CARD NUMBER 1: The description of the case is put on this card.

CARD NUMBER 2:

NO	-	Problem number.
MNMAX	-	Maximum number of Fourier terms in applied load; limited to 10.
IGEOM	-	When the dimensional radius is defined by: 1. Input data on cards, IGEOM < 0 2. Programmed instructions*, IGEOM \geq 0
ISTIF	-	When the dimensional stiffnesses B and D are defined by: 1. Input data on cards, ISTIF < 0 2. Programmed instructions*, ISTIF \geq 0
ILOAD	-	When the dimensional loads are defined by: 1. Input data on cards, ILOAD < 0 2. Programmed instructions*, ILOAD \geq 0
IBCINL	-	When the shell: 1. Has an initial pole, IBCINL < 0 2. Does not have an initial pole, IBCINL \geq 0
IBCFNL	-	When the shell: 1. Has a final pole, IBCFNL < 0 2. Does not have a final pole, IBCFNL \geq 0
KMAX	-	Maximum number of meridian stations; limited to 20.

*See Optional Programming Requirements.

CARD #1 FORMAT (12A6)

0 Sample Problem 10 - Unsymmetrically loaded spherical cap, test case.

72

DESCRIPTION

CARD #2 FORMAT (14I5)
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70
0 10 31 0 1 0 0 -1 0 1 7 1 4 1 4 1 1 4 1
MINMAX TFROM TSTIF TLOAD TBCFL KMAX IPREQ NTHMAX MAXM LSMAX UCHMAX ITRMMAX
NO.

CARD #3 FORMAT (6E12.8)
0 .3 12 1 1000. 24 30.E6 36 1.0 48 1000.0 60 105.
1 NU SIGO ELAST TKN CHAR S
DELOAD EPS

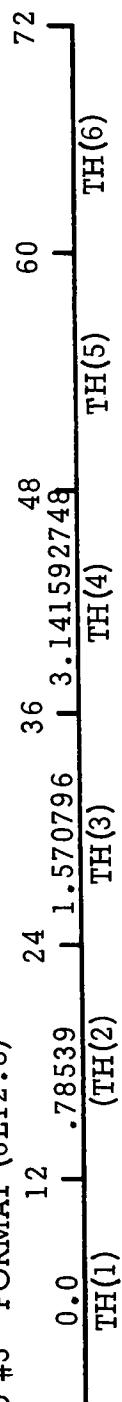
CARD #4 FORMAT (6E12.8)
0 .2 12 1 .01 24
1 DELOAD EPS

64

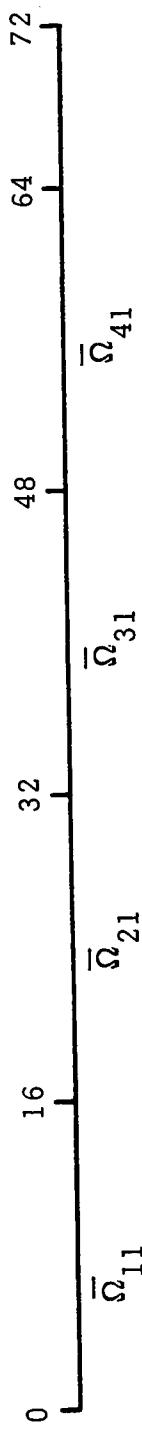
FIGURE 12. Input Format

72

CARD #5 FORMAT (6E12.8)



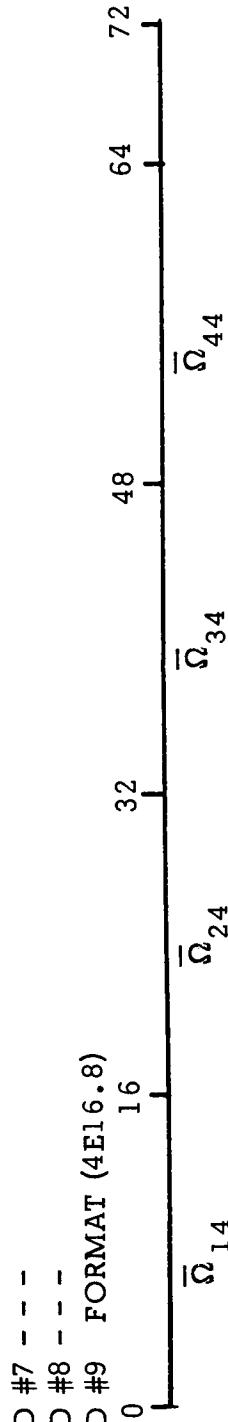
CARD #6 FORMAT (4E16.8)



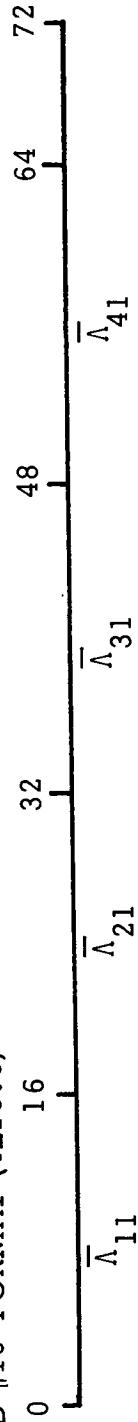
CARD #7 -- -

CARD #8 -- -

CARD #9 FORMAT (4E16.8)



CARD #10 FORMAT (4E16.8)



CARD #11-- -

CARD #12-- -

FIGURE 12. Continued

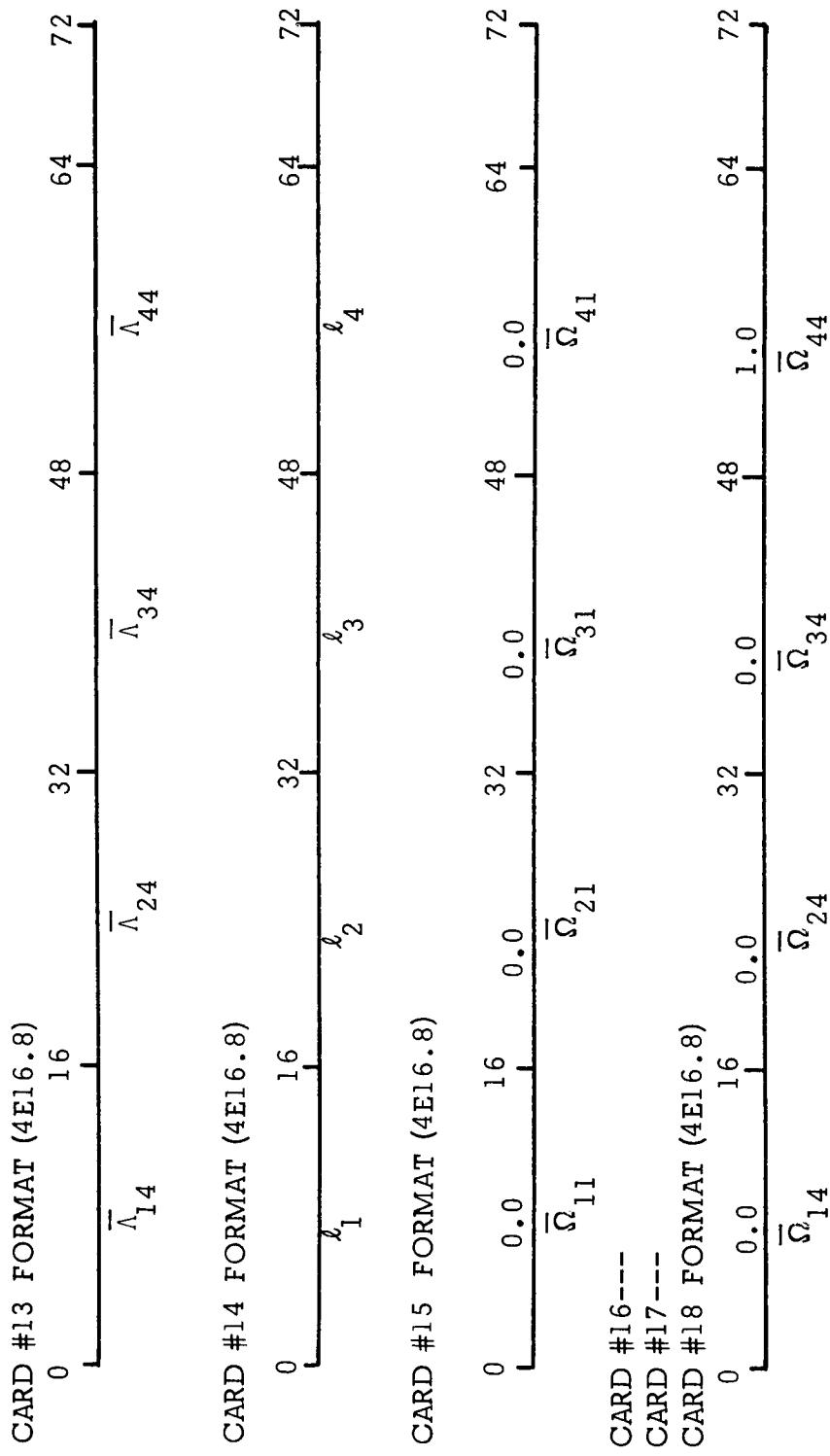
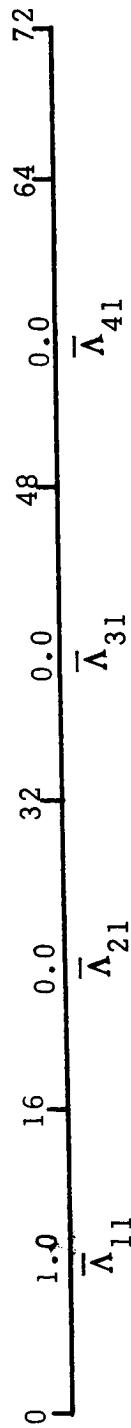
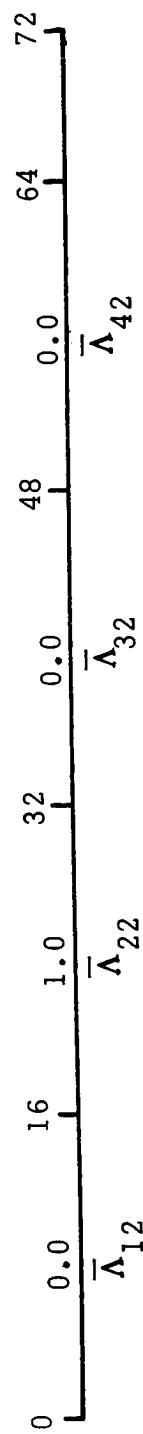


FIGURE 12. Continued

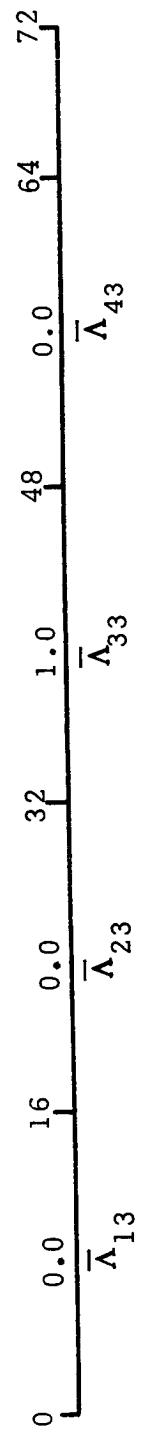
CARD #19 FORMAT (4E16.8)



CARD #20 FORMAT (4E16.8)



CARD #21 FORMAT (4E16.8)



CARD #22 FORMAT (4E16.8)

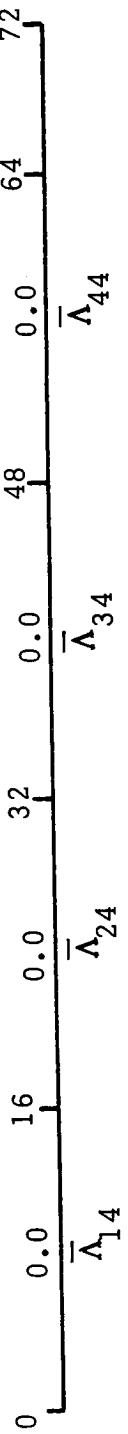
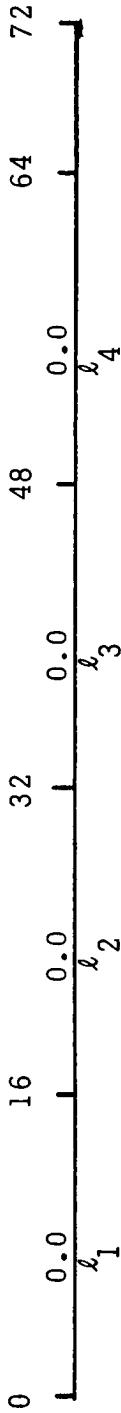
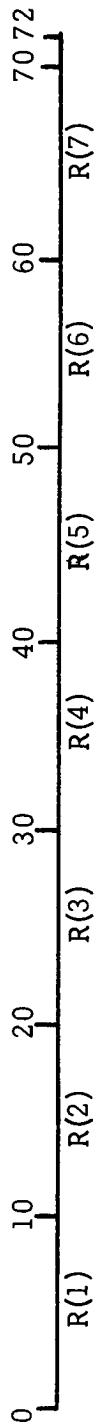


FIGURE 12 . Continued

CARD #23 FORMAT (4E16.8)

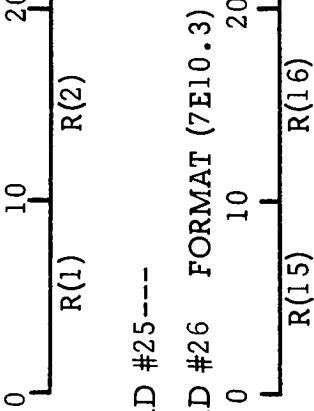


CARD #24 FORMAT (7E10.3)



CARD #25---

CARD #26 FORMAT (7E10.3)



CARD #27-29 FORMAT (7E10.3)



CARDS #30-32 FORMAT (7E10.3)



FIGURE 12. Continued

CARD #33 FORMAT (I4)

0 4
 NM(MN=1)

CARD #34-36 FORMAT (7E10.3)



CARD #37-39 FORMAT (7E10.3)



CARD #40-42 FORMAT (7E10.3)



CARD #43-45 FORMAT (7E10.3)



CARD #46-48 FORMAT (7E10.3)



For the test case thirty-two additional blank cards are required for the input load data for MN=2 and MN=3

FIGURE 12. Continued

APPENDIX E

- IFREQ - Output control; determines the stations to be printed; IFREQ, 2 x IFREQ, 3 x IFREQ, ...
- NTHMAX - Output control; number of positions of printout around the shell circumference; limited to 6.
- MAXM - Maximum number of Fourier terms in the solution; limited to 10.
- LSMAX - Maximum number of load steps; the program transfers to the next case when the load step equals LSMAX.
- LCHMAX - Maximum number of DELOAD reductions by a factor of 5; the program transfers to the next case when the number of iterations exceeds ITRMAX and the number of previous DELOAD reductions equals LCHMAX; recommended value, 3.
- ITRMAX - Maximum number of iterations; recommended value, 20-40.

CARD NUMBER 3:

- NU - Poisson's ratio.
- SIGO - σ_0 ; reference stress level used to nondimensionalize input data for computations.
- ELAST - E_0 ; reference elastic modulus used to nondimensionalize input data for computations.
- TKN - h_0 ; reference thickness used to nondimensionalize input data for computations.
- CHAR - a ; reference length used to nondimensionalize input data for computations.
- S - Total dimensional length of shell meridian.

APPENDIX E

CARD NUMBER 4:

- DELOAD - Incremental load factor; after each solution is obtained the applied load and the boundary matrix ℓ are increased by the amount DELOAD \times input value; recommended value, .2.
- EPS - Convergence criterion; convergence occurs when the difference between the solutions for two consecutive iterations is smaller than EPS times the latest solution*; recommended value, .01.

CARD NUMBER 5: The list of angles, in radians, NTHMAX values, where a printout of the summation of the Fourier terms of the solution is desired; this card is blank if NTHMAX is zero.

CARD NUMBER 6 through - CARD NUMBER 14 These cards specify the initial boundary conditions; cards 6, 7, 8, and 9 specify the elements of the $\bar{\Omega}$ matrix; cards 10, 11, 12, and 13 specify the elements of the $\bar{\Lambda}$ matrix; card 14 specifies the elements of the ℓ column matrix; see equation (16) for the definition of $\bar{\Omega}$, $\bar{\Lambda}$, and ℓ ; these cards are blank if the shell has an initial pole.

CARD NUMBER 15 through - CARD NUMBER 23 These cards specify the final boundary conditions; cards 15, 16, 17, and 18 specify the elements of the $\bar{\Omega}$ matrix; cards 19, 20, 21, and 22 specify the elements of the $\bar{\Lambda}$ matrix; card 23 specifies the elements of the ℓ column matrix; see equation (16) for the definition of $\bar{\Omega}$, $\bar{\Lambda}$, and ℓ ; these cards are blank if the shell has a final pole.

CARD NUMBER 24 through - CARD NUMBER 26 These cards specify the dimensional radius r for each station; all three cards must always be used even if KMAX < 15; these cards are blank if r is to be input analytically**.

*Provision has been made to bypass comparison of solution components whose nondimensional value is less than 1×10^{-6}

**See Optional Programming Requirements.

APPENDIX E

CARD NUMBER 27 through **CARD NUMBER 29** These cards specify the dimensional inplane stiffness B for each station; see equation (11a) for the definition of B; all three cards are blank if B is to be input analytically*.

CARD NUMBER 30 through **CARD NUMBER 32** These cards specify the dimensional bending stiffness D for each station; see equation (11b) for the definition of D; all three cards must be used even if KMAX < 15; these cards are blank if D is to be input analytically*.

CARD NUMBER 33 through **CARD NUMBER 48** These 16 cards specify one Fourier index and the Fourier coefficients of the dimensional loads associated with that index; card number 33 specifies one Fourier index N; card numbers 34, 35, and 36 specify $\sigma_o h_o p^{(N)} / a$ for each meridional station; card numbers 37, 38, and 39 specify $\sigma_o h_o p_s^{(N)} / a$ for each meridional station; card numbers 40, 41, and 42 specify $\sigma_o h_o p_\theta^{(N)} / a$ for each meridional station; card numbers 43, 44, and 45 specify $\sigma_o h_o (1-\nu) t_T^{(N)} / a$ for each meridional station; card numbers 46, 47, and 48 specify $\sigma_o h_o^3 (1-\nu) m_T^{(N)} / a$ for each meridional station; see equations (23) for the definition of $p^{(N)}$, $p_s^{(N)}$, and $p_\theta^{(N)}$ and equations (24 and 34) for the definition of $t_T^{(N)}$ and $m_t^{(N)}$.

CARD NUMBER 49 through **CARD NUMBER 64** An additional corresponding set of 16 cards must be prepared for each value of the Fourier index n being considered; thus, there will be a total of 16 x MNMAX load cards; all 16 x MNMAX cards will be blank if the loads are to be specified by programming instructions*.

CARD NUMBER 65 through **CARD NUMBER 80,**
etc.

*See Optional Programming Requirements.

APPENDIX E

- Important Notes:
1. The alternate input-output features of the IBJOB monitor (ALTIO) is required.
 2. Because of the method used in referencing data in COMMON, the program must be compiled in the order listed in Appendix F.

Optional Programming Requirements

Subroutines GEOM, STIF and PTLOAD, i.e., decks 2, 3 and 4, are used to set up the initial conditions for the shell geometry stiffness and applied loads. The initial data can either be computed or read into the program with arrays of numbers. Each of these subroutines contains a fixed point argument. The arguments are IGEOM, ISTIF and ILOAD respectively. In each case, a negative value indicates the input data read in on cards will be used and a non-negative value indicates initial values are to be computed from an analytical model.

In subroutine GEOM, FORTRAN statements for the quantities r , $\frac{r'}{r}$, $1/R_\theta$, $1/R_s$, and $(1/R_s)'$ must be provided if IGEOM is non-negative. In order to program an analytical shell the following list shows the correspondence between the FORTRAN variables and the problem variables:

R(K)	-	r
GAM(K)	-	r'/r
OMT(K)	-	$1/R_\theta$
OMXI(K)	-	$1/R_s$
DEOMXI(K)	-	$(1/R_s)'$

The subscript K ranges from 1 to KMAX, the maximum number of meridian stations. Figure 13 shows an example of a sequence of instructions contained in subroutine GEOM for computing the initial data for the spherical cap test case. Note that the first FORTRAN statement is numbered 50 and that the transfer statement is GO TO 1000. This must always be so.

Subroutine STIFF uses the indicator ISTIF. If ISTIF is non-negative, the following dimensional variables are to be initialized in the computational loop for K = 1, KMAX:

B(K)	-	membrane stiffness B
DB(K)	-	derivative of membrane stiffness, B'

APPENDIX E

```
IF (IGEOM) 500, 50, 50
```

```
50 DO 10 K = 2, KMAX
```

```
RK = K
```

```
THET = (RK - 1.0)*DEL/1000.
```

```
R(K) = SIN(THET)*1000.
```

```
GAM(K) = COS (THET) / R(K)
```

```
OMT(K) = 1.0 / 1000.
```

```
OMXI(K) = 1.0 / 1000.
```

```
10 DEOMXI(K) = 0.0
```

```
R(1) = 0.0
```

```
GAM(1) = 0.0
```

```
OMT(1) = 1.0 / 1000.
```

```
OMXI(1) = 1.0 / 1000.
```

```
DEOMX(1) = 0.0
```

```
GO TO 1000
```

```
500 RADIUS values for each meridian station have been read in.  
Compute GAM(K) , OMT(K) , OMXI(K) , and DEOMXI(K) using  
finite difference formulae for each meridian station.
```

```
1000 WRITE (6,102)  
Statement 1000 and the following write out the initial data  
describing the shell geometry.
```

Inserted FORTRAN
Statements for
Analytical Model

FIGURE 13. FORTRAN Statements for Subroutine GEOM

APPENDIX E

$D(K)$ - bending stiffness, D

$DD(K)$ - derivative of bending stiffness, D'

The first FORTRAN statement must be statement number 50 and the computations for the analytical model must be followed by a GO TO 1000 statement in order to print out the input data and to return properly. The source program for the test case computes the B , D , DB , and DD arrays.

Subroutine PTLOAD uses the indicator ILOAD. If ILOAD is nonnegative the following dimensional variables must be initialized in the computational loop:

$N(MN)$ $MN = 1, MN MAX$ - array of Fourier numbers.

$PR(K, MN)$	-	$\frac{\sigma_o h_o}{a} p(n)$	$K = 1, KMAX$ for $MN = 1, MN MAX$
$PT(K, MN)$	-	$\frac{\sigma_o h_o}{a} p_\theta(n)$	
$PX(K, MN)$	-	$\frac{\sigma_o h_o}{a} p_s(n)$	
$TT(K, MN)$	-	$\sigma_o h_o (1 - \nu) t_T^{(n)}$	
$MT(K, MN)$	-	$\sigma_o h_o^3 (1 - \nu) m_T^{(n)} / a$	
$DTT(K, MN)$	-	Derivative of TT array	
$DMT(K, MN)$	-	Derivative of MT array	

The first FORTRAN statement must be numbered 50 and the computations for the analytical model must be followed by a GO TO 1000 statement in order to print out the input data and return properly.

The dimensional incremental length DEL , the number of meridian stations, $KMAX$ and length of the meridian, S are FORTRAN variables available for computations through COMMON. These variables may be needed in order to compute a given analytical model.

APPENDIX E

Output Data

The computer program prints out dimensional data corresponding to the dimensional data read into the program. Subroutine OUTPUT converts the solution from nondimensional computational form into the proper formated dimensional form for output. The first output from the program consists of most of the input data contained on cards 1 - 23. Next, the following geometric quantities are listed as a function of the station*:

RADIUS	-	r
GAMMA	-	r'/r
OMEGA S	-	$1/R_s$
OMEGA THETA	-	$1/R_\theta$
DEOMEGA S	-	$(1/R_s)'$

Following this comes the inplane and bending stiffness quantities:

B	-	B
D	-	D
DB	-	B'
DD	-	D'

as a function of the station. The applied loads are listed next as a function of station

*A prime denotes differentiation with respect to the meridional coordinate s .

APPENDIX E

N	- Fourier index, n
P	- $\frac{\sigma_o h}{a} p(n)$
PS	- $\frac{\sigma_o h}{a} p_s(n)$
PT	- $\frac{\sigma_o h}{a} p_\theta(n)$
TT	- $\sigma_o h (1 - \nu) t_T^{(n)}$
MT	- $\sigma_o h^3 (1 - \nu) m_T^{(n)} / a$
DTT	- $\sigma_o h (1 - \nu) t'_T^{(n)}$
DMT	- $\sigma_o h^3 (1 - \nu) m'_T^{(n)} / a$

Finally, after each solution is obtained the following output is listed:

LSTEP	- The load step
ALOAD	- The proportion of the input load acting on the shell
ITR	- The number of iterations required to obtain the solution

and at each station there is printed MODAL OUTPUT followed by

N	- Fourier index, n
U	- $\frac{a \sigma_o}{E_o} u(n)$
V	- $\frac{a \sigma_o}{E_o} v(n)$
W	- $\frac{a \sigma_o}{E_o} w(n)$
PHIS	- $\frac{\sigma_o}{E_o} \varphi_s(n)$
PHITH	- $\frac{\sigma_o}{E_o} \varphi_\theta(n)$

APPENDIX E

PHI	-	$\frac{\sigma_o}{E_o} \varphi^{(n)}$
MS	-	$\frac{\sigma_o h_o^3}{a} m_s^{(n)}$
MTH	-	$\frac{\sigma_o h_o^3}{a} m_\theta^{(n)}$
MSTH	-	$\frac{\sigma_o h_o^3}{a} m_{s\theta}^{(n)}$
TS	-	$\sigma_o h_o t_s^{(n)}$
TTH	-	$\sigma_o h_o t_\theta^{(n)}$
TSTH	-	$\sigma_o h_o t_{s\theta}^{(n)}$
QS	-	$\sigma_o h_o f_s^{(n)}$

and then THETA OUTPUT followed by

THETA	-	θ
U	-	U
V	-	V
W	-	W
PHIS	-	Φ_s
PHITH	-	Φ_θ
PHI	-	Φ
MS	-	M_s

APPENDIX E

MTH	-	M_θ
MSTH	-	$M_{s\theta}$
NS	-	N_s
NTH	-	N_θ
NSTH	-	$N_{s\theta}$
QS	-	Q_s

The execution for each case terminates and the program transfers to the next case when either the load step equals LSMAX or the number of iterations exceeds ITRMAX and the number of previous DELOAD reductions equals LCHMAX. If the former occurs, the statement

END PROBLEM NUMBER - LSTEP = LSMAX

is printed; if the latter occurs

END PROBLEM NUMBER - LCHANG = LCHMAX

is printed. When the number of iterations exceeds ITRMAX and the number of previous DELOAD reductions is less than LCHMAX, the statement

CONVERGENCE FAILURE, STEP SIZE REDUCED

is printed.

Figure 14 shows a portion of the output obtained from the sample case.

Subroutine Descriptions

Flow charts showing the orderly progression of the logic for each of the subroutines are shown in Appendix G. Note that whenever K appears in the argument of a subroutine, it refers to each meridional station, not just the final station.

PROBLEM NUMBER 10

SAMPLE PROBLEM 10 -- UNSYMMETRICALLY LOADED SPHERICAL CAP, TEST CASE

BOUNDARY CONDITIONS

INITIAL POLE

FINAL EDGE OMEGA BAR

INITIAL POLE	FINAL EDGE	OMEGA BAR	LAMBDA BAR
0.	0.	0.	1.0000E 00
0.	0.	0.	0.
0.	0.	0.	1.0000E 00
0.	0.	0.	0.

INPUT DATA

80

LOAD INCREMENT = 0.200
MAXIMUM NUMBER OF LOAD STEPS = 2
MAXIMUM NUMBER OF ITERATIONS = 4C
POISSONS RATIO = 0.30000
CHARACTERISTIC SHELL DIMENSION = C.10000E-04
REFERENCE PLASTICITY = 0.30000E 08
THETA 0.
CONVERGENCE CRITERION = C.0100
MAXIMUM NUMBER OF MODES = 4
NUMBER OF LOAD INCREMENT CHANGES = 1
REFERENCE THICKNESS = 0.10000E 01
REFERENCE STRESS = 0.10000E 04
MERIDIAN LENGTH = 0.10500E 03
0.78539800E 00 0.15707960E 01 0.31415927E 01

Figure 14. Sample Output

STATION	RADIUS	GAMMA	OMEGA S	OMEGA THETA	DEOMEGA S
1	0.	0.5714E-01	0.1000E-02	0.	
2	0.1750E 02	0.2850E-01	0.1000E-02	0.	
3	0.3499E 02	0.1933E-01	0.1000E-02	0.	
4	0.5246E 02	0.1426E-01	0.1000E-02	0.	
5	0.6994E 02	0.1140E-01	0.1000E-02	0.	
6	0.8739E 02	0.9489E-02	0.1000E-02	0.	
7	0.1048E 03	0.9489E-02	0.1000E-02	0.	
STATION	R	D	DR	DD	
1	0.2730E 08	0.2275E 07	0.		
2	0.2730E 08	0.2275E 07	0.		
3	0.2730E 08	0.2275E 07	0.		
4	0.2730E 08	0.2275E 07	0.		
5	0.2730E 08	0.2275E 07	0.		
6	0.2730E 08	0.2275E 07	0.		
7	0.2730E 08	0.2275E 07	0.		
STATION	P	PS	PT	DTT	DMT
1	0	-0.1500E 02	0.	0.	0.
2	0	-0.1500E 02	0.	0.	0.
3	0	-0.1500E 02	0.	0.	0.
4	0	-0.1500E 02	0.	0.	0.
5	0	-0.1500E 02	0.	0.	0.
6	0	-0.1500E 02	0.	0.	0.
7	0	-0.1500E 02	0.	0.	0.
1	1	-0.1910E 02	0.	0.	0.
2	1	-0.1910E 02	0.	0.	0.
3	1	-0.1910E 02	0.	0.	0.
4	1	-0.1910E 02	0.	0.	0.
5	1	-0.1910E 02	0.	0.	0.
6	1	-0.1910E 02	0.	0.	0.
7	1	-0.1910E 02	0.	0.	0.
1	3	0.6370E 01	0.	0.	0.
2	3	0.6370E 01	0.	0.	0.

Figure 14. Sample Output (Continued)

LSTEP= 2
 ALNAD=0.400
 ITTR= 7

INITIAL POLE						
N	U	V	W	PHIX	PHIY	PHITH
0	-0.	-0.	-1.2116E-01	0.	0.	0.
1	-1.1334E-02	1.1334E-02	0.	1.0633E-02	-1.0633E-02	
3	0.	0.	0.	0.	0.	
2	-0.	-0.	-0.	0.	0.	

STATION 1- MODAL OUTPUT						
N	U	V	W	PHIX	PHIY	PHITH
0	3.5046E-04	-0.	-1.2116E-01	4.3826E-05	-0.	
1	-1.1334E-02	1.2519E-02	-1.5200E-01	6.7158E-03	-8.6737E-03	
3	1.6844E-04	-1.7050E-04	4.2294E-03	-6.4467E-04	7.2490E-04	
2	-3.8431E-04	4.3695E-04	-2.2295E-03	2.0645E-04	-2.5438E-04	

STATION 2- MODAL OUTPUT						
N	U	V	W	PHIX	PHIY	PHITH
0	1.4368E-01	1.0007E-01	0.	-3.2143E-03	-3.2077E-03	
1	-6.4337E-02	-4.4769E-02	1.9572E-02	-2.8084E-03	-3.1129E-03	
3	-4.9577E-01	1.8405E-02	1.2093E-02	3.9024E-02	-3.3216E-02	
2	9.7075E-00	-3.6394E-01	-1.5858E-01	-1.2768E-02	6.5449E-01	

STATION 2- THETA OUTPUT						
THETA	U	V	W	PHIX	PHIY	PHITH
0	-1.1199E-02	0.	-2.7116E-01	6.3214E-03	-0.	
1	-7.7831E-03	9.1689E-03	-2.3163E-01	5.2484E-03	-5.8750E-03	
3	7.3476E-04	1.2690E-02	-1.1893E-01	-1.6263E-04	-9.3986E-03	
2	1.1132E-02	-6.4669E-10	2.4384E-02	-5.8208E-03	3.6679E-10	

Figure 14. Sample Output (Continued)

APPENDIX E

DECK 1 - MAIN

This program controls the logical connections between the subroutines. The case description, control parameters, physical constants, and boundary conditions are both read and printed out in this routine. The boundary conditions are nondimensionalized and many of the common indices and coefficients are determined here. The iteration procedure, the test for convergence, the load incrementing procedure, and the calculation for the estimate of the next solution are all carried out here.

DECK 2 - Subroutine GEOM

This subroutine prepares the geometry functions of the shell. The user may either employ cards to input the radius r or he may prepare special instructions for the radius r , the radii of curvature $1/R_s$, and $1/R_\theta$, the derivative $\frac{d}{ds}(1/R_s)$, and the function $\frac{1}{r} \frac{dr}{ds}$. If the card input option for r is used, the other functions are automatically calculated within the routine. All input functions are dimensional and the print out is also in dimensional form. The non-dimensional functions p , γ , w_θ , w_s , and $\frac{dw_s}{ds}$ are automatically calculated within the routine. All derivatives, including edge derivatives, are computed using finite difference formulae with an error $O(\Delta^2)$.

DECK 3 - Subroutine STIFF

This subroutine prepares the inplane and bending stiffnesses of the shell. The user may either employ cards to input the stiffnesses B and D or he may prepare special instructions for B , D , $\frac{dB}{ds}$ and $\frac{dD}{ds}$. If the card input option for B and D is used, $\frac{dB}{ds}$ and $\frac{dD}{ds}$ are calculated automatically within the routine using finite difference formulae with an error $O(\Delta^2)$ for all derivatives, including edge derivatives. All input functions are dimensional and all printout is also in dimensional form. The nondimensional functions b , d , $\frac{db}{ds}$, and $\frac{dd}{ds}$ are automatically calculated within the routine.

APPENDIX E

DECK 4 - Subroutine PTLOAD

This subroutine prepares the pressure and thermal loads applied to the shell. The user may either employ cards to input the Fourier indices and the Fourier coefficients of the loads $\sigma_o h_o p^{(n)}/a$, $\sigma_o h_o p_s^{(n)}/a$, $\sigma_o h_o p_\theta^{(n)}/a$, $\sigma_o h_o (1-\nu)t_T^{(n)}$, and $\sigma_o h_o^3 (1-\nu)m_T^{(n)}$ or he may prepare special instructions for the above loads plus the derivatives $\sigma_o h_o (1-\nu) \frac{d}{ds} t_T^{(n)}$ and $\sigma_o h_o^3 (1-\nu) \frac{d}{ds} m_T^{(n)}$. If the card input option for the Fourier indices and load coefficients is used, the derivatives are automatically calculated within the routine using finite difference formulae with an error $O(\Delta^2)$ for all derivatives, including edge derivatives. All input functions are dimensional and all printout is also in dimensional form. The nondimensional functions $p^{(n)}$, $p_s^{(n)}$, $p_\theta^{(n)}$, $t_T^{(n)}$, and $m_T^{(n)}$ are automatically calculated within the routine.

DECK 5 - Subroutine ACOEFF

This subroutine prepares the set of coefficients that make up the E, F, G and e matrices given by equation (36). The elements of E, F, G and e, as given in Appendix C, can be expressed in terms of polynomials of n and this subroutine computes the coefficients of these polynomials.

DECK 6 - Subroutine PMATRX

This subroutine calls subroutines HJ, EFG, ABC, and PANDD to set up the P matrix given by equations (48a and 49a) for each meridian station. Matrices DL, DG, and DF are set up for the calculation of X_1 given by equation (48b), where

$$X_1 = DL\ell_1 + DGg_1 + DFf_1$$

The special P matrix for a shell with an initial pole, given in Appendix D, is also computed here. Matrices ZF1M, ZF2M, ZF3M, and ZF4M are set up for the calculation of Z_{K+1} given by equation (51) where

$$Z_{K+1} = ZF1M\ell_K + ZF2MX_K + ZF3MX_{K-1} + ZF4Mf_K$$

APPENDIX E

If the shell has a final pole, the matrices CL0, CLI and CL2 are prepared for the calculation of Z_K given by equation (D-3) where

$$Z_K = CL0X_{K-1} \text{ or } CLI X_{K-1} \text{ or } CL2 X_{K-1}$$

depending upon whether $n = 0, 1$ or 2 .

DECK 7 - Subroutine HJ(K, MN)

This subroutine computes the elements of the H and JAY matrices given by equation (40) for both boundaries of the shell. The elements of H and JAY are defined in Appendix C.

DECK 8 - Subroutine EFG(K, MN)

This subroutine prepares the elements of the E, F, and G matrices using the coefficients computed in subroutine ACOEFF for each meridian station K and for each Fourier mode MN. The matrices E, F, and G are given by equation (36) and the elements are defined in Appendix C.

DECK 9 - Subroutine ABC

This subroutine computes the elements of the A, BEE, and C matrices defined by equation (43).

DECK 10 - Subroutine PANDD(K, MN)

This subroutine computes the elements of the P, DEE, and DST matrices for each meridian station K and Fourier mode MN. The matrix P is given by equation (49a), and the DEE and DST matrices are defined by

$$X_1 = DEEg_1 - DSTX_{1-1}$$

where X_1 is given by equation (49b). These matrices are computed and saved because they do not change during either the iteration procedure or the load increment procedure, i.e., they are a function of the shell's initial geometry and stiffness.

APPENDIX E

DECK 11 - Subroutine XANDZ

This subroutine computes the X vector using the P , DEE and DST matrices and solves for the Z vector for the applied and pseudo loads. The matrices $PHIBET$ and $TEAETA$ are called and the previous solution for Z , or the estimated value of Z , is used to calculate the nonlinear Beta and Eta terms. The matrices FFS and FLS are the values of f at the initial and final edges of the shell as given by equation (40) and equations (C-59 through C-62) in Appendix C. The subroutine $FORCE(K)$ is called to calculate the load vector g , (GEE), and the X vector at each meridian station. Once the X vector is obtained for all meridian stations the solution for Z_{K-1} given by equation (51) is obtained, and the solution for Z_1 at all the other meridian stations defined by equation (47) is obtained. The solution Z at the imaginary station off the initial edge of the shell is obtained last. The test for convergence of the solution is made as Z is computed. The special conditions for computing Z at either an initial or a final pole are also in this routine.

DECK 12 - Subroutine INLPOL

This subroutine computes, for a shell with an initial pole, the nonlinear terms β_s , β_θ , $\beta_{s\theta}$, η_{ss} , and $\eta_{\theta s}$ at the pole. The appropriate equations are derived in Appendix D.

DECK 13 - Subroutine FNLPOL

This subroutine computes, for a shell with a final pole, the nonlinear terms β_s , β_θ , $\beta_{s\theta}$, η_{ss} , and $\eta_{\theta s}$ at the pole. The appropriate equations are derived in Appendix D.

DECK 14 - Subroutine MODES

In MODES, arrays that define those sets of indices that combine to equal each value of n in the problem are determined. MODES is called prior to the first iteration and after every iteration until a specified number of Fourier terms is reached. Each Fourier index in the problem is subtracted from all other Fourier indices and the result is compared with all Fourier indices to see if the new value exists in the program*.

*The same comparison is never made twice.

APPENDIX E

If it does, the locations of the two indices that made the combination are stored in two special two-dimensional arrays, ID and JD. One argument of each array is the value of the new index and the other is the number of combinations of indices that also give this value of the index*. If there is no index in the program that matches the new one, then a new Fourier term has been generated and will be considered in the next iteration for solution. The variable MAXD stores the total number of such combinations for each value of the Fourier index. In a similar manner, each index is added to every other index and the sum compared with all indices. This result is stored in the two two-dimensional arrays, IS and JS, in the same manner as was done for the subtraction case. The variable MAXS stores the total number of summation combinations for each value of the Fourier index. A special routine handles the cases where the index is added to and subtracted from itself. The two-dimensional array IJS stores the location of the index and the variable MAXSY stores the total number of such combinations. With this procedure the series of products that make up the β 's and η 's contain no zero terms, and the summation is carried out in PHIBET and TEAETA over specifically defined limits.

DECK 15 - Subroutine OUTPUT

This subroutine prepares the printout material. It is called when either the solution has converged at a load step or when the number of iterations equals the maximum allowable and the number of load changes equals the maximum allowable. In the latter case, the latest value of Z is printed and the problem ends. The Fourier coefficients of the inplane forces, meridional transverse force, circumferential bending moment, twisting moment and rotations are computed and printed with the solution Z for the Fourier coefficients of the three displacements and meridional bending moment. This output material is converted from dimensionless form to dimensional form here. Provision is made to print at only selected meridional stations. This subroutine also performs the summation process for computing the total values of the forces, moments, displacements, and rotations at the Theta positions around circumference prescribed in the input data. Special sections for printing the solution at an initial and final pole are also included here.

DECK 16 - Subroutine PHIBET(K)

This subroutine calculates the Phis and carries out the multiplying and summation procedure for computing the Betas derived in Appendix B for a given meridional station K. The arrays IS, JS, ID, JD, IJS, MAXS, MAXD, and MAXSY prepared in subroutine MODES are used here.

*The same comparison is never made twice.

APPENDIX E

DECK 17 - Subroutine TEAETA(K)

This subroutine calculates the inplane forces and carries out the multiplying and summation procedure for computing the Etas derived in Appendix B for a given meridional station K. The arrays IS, JS, ID, JD, IJS, MAXS, MAXD, and MAXSY prepared in subroutine MODES are used here.

DECK 18 - Subroutine FORCE(K)

This subroutine computes the GEE vector, equation (43), and the X vector, equation (49b), for a given meridional station K. The arrays CEEE1 through CEEE4 are the linear values of g based upon the full input values of the applied pressures and temperatures. The vector GEE is first the linear value of g for the current load step and then becomes the nonlinear value of g. The array GEES is the nonlinear value of g at station l.

DECK 19 - Subroutine UPDATE

This subroutine updates the storage locations of the Betas and Etas. It is called in subroutine XANDZ after a meridian station change.

DECK 20 - Subroutine MATINV (A, N, B, M, DETERM, IPIVOT, INDEX, NMAX, ISCALE)

This subroutine solves the matrix equation $AX = B$ where A is a square coefficient matrix and B is a matrix of constant vectors. A^{-1} is also obtained and the determinant of A is available. Jordan's method is used to reduce a matrix A to the identity matrix I through a succession of elementary transformations: $t_n, t_{n-1}, \dots, t_1, A = I$. If these transformations are simultaneously applied to I and to a matrix B of constant vectors, the result is A^{-1} and X where $AX = B$. Each transformation is selected so that the largest element is used in the pivotal position. The subroutine has been compiled with a variable dimension statement A(NMAX, NMAX), B(NMAX, M). The following must be dimensioned in the calling program: IPIVOT(NMAX), INDEX(NMAX, 2), A(NMAX, NMAX), B(NMAX, M) where IPIVOT and INDEX are temporary storage blocks. An overflow may be caused by a singular matrix. The definition of the arguments of this subroutine are as

APPENDIX E

follows:

A = first location of a 2-dimensional array of the A matrix.

N = location of order of A;

$$1 \leq N \leq NMAX$$

B = first location of a 2-dimensional array of the constant vectors B.

M = location of the number of column vectors in B. M = 0 signals that the subroutine is to be used solely for inversion, however, in the call statement an entry corresponding to B must still be present.

DETERM - gives the value of the determinant by the following formula:

$$\text{DET}(A) = (10^{18})^{\text{ISCALE}}(\text{DETERM})$$

IPIVOT - temporary storage block.

INDEX - temporary storage block.

NMAX = location of maximum order of A as stated in dimension statement of calling program.

ISCALE - used in obtaining the value of the determinant by the following formula:

$$\text{DET}(A) = (10^{18})^{\text{ISCALE}}(\text{DETERM})$$

At the return to the calling program A^{-1} is stored at A and X is stored at B.

APPENDIX F

PROGRAM LISTING

Important Notes

1. The alternate input-output feature of the IBJOB monitor (ALTIO) is required.
2. Because certain liberties have been taken in referencing data in COMMON the program must be compiled in the order listed.

C MAIN--THIS PROGRAM IS ENTERED FOR THE START OF EACH PROGRAM GROUP.
 REAL NU,LAM,LAM2
 COMMON /IBL9/MAXM/IBL1/MNMAX/IBL5/IBL5/IBCINL,IBCFNL/IBL4/KMAX,KL/IBL
 110/IFREQ,NTHMAX /IBL12/KMAX2,NCONV/IBL6/KLL/IBL
 28/LSTEP,ITR/IBL11/ICORFL,IPASS/IBL2/N(10),MNINIT/BL16/EPS
 3 /BL17/DEL/BL18/EL1(4),ELL(4)/BL13/OMEG1(4,4),CAPL1(4,4)*OMEGL(4,4),
 CAPLL(4,4)*UNIT(4,4)/BL19/TH(6)/BL6/Z(4,22,10),SOE,OSE,ALOAD
 5/BL12/TDL1,TDEL/BL7/D1,S1/BL31/DELSQ
 COMMON /IBL7/MNMAXO,MAXD(10),MAXS(10),MAXY(10),IS(5,10),JS(5,10),
 IID(10,10),JD(10,10),IJS(10)/BL15/NU,U1(10),V1(10),W1(10),U2(10),
 2V2(10),W2(10),U3(10),V3(10),W3(10)/BL14/LAM2,LSD18,LSD1N
 COMMON /BL21/ZQ(4,25,10)
 DIMENSION CONST(20),DESCR(12)
 DIMENSION SIGT(2),SIGC(2)
 500 READ(5,100)DESCR
 READ(5,101)NO,MNMAX,IGEOM,ISTIF,ILOAD,IBCINL,IBCFNL,KMAX,
 1 IFREQ,NTHMAX,MAXM,LSMAX,LCHMAX,ITRMAX
 READ(5,102)NU,SIGO,ELAST,TKN,CHAR,S,DELOAD,EPS
 KL=KMAX-1
 KLL=KMAX-2
 KMAX1=KMAX+1
 KMAX2=KMAX+2
 AK=KL
 DEL=S/AK
 DELSQ=DEL**2
 TDL1=.5/DEL
 TDEL=2.*DEL
 SIGT(1)=SIGO*TKN
 SIGT(2)=SIGO/ELAST
 SIGC(1)=SIGO*CHAR/ELAST
 SIGC(2)=SIGO*TKN**3/CHAR
 READ(5,105)(TH(NTH),NTH=1,NTHMAX)
 3 READ(5,103)OMEG1,CAPL1,EL1
 7 READ(5,103)OMEGL,CAPLL,ELL

```

10 WRITE(6,201)NO
  WRITE(6,202)DESCR
  WRITE(6,203)
  IF(IBCINL.LT.0) GO TO 11
  WRITE(6,204)
  WRITE(6,205) ((OMEG1(I,J),J=1,4),(CAPL1(I,J),J=1,4),ELL(I),I=1,4)
  DO 98 I = 1,4
  DO 98 J = 1,4
  KKLM = J/4 + 1
  OMEG1(I,J) = OMEG1(I,J)*SIGT(KKLM)
98   CAPL1(I,J) = CAPL1(I,J)*SIGC(KKLM)
14   IF(IBCFCNL.LT.0) GO TO 12
  WRITE(6,224)
  WRITE(6,205) ((OMEGL(I,J),J=1,4),(CAPLL(I,J),J=1,4),ELL(I),I=1,4)
  DO 99 I = 1,4
  DO 99 J = 1,4
  KKLM = J/4 + 1
  OMEGL(I,J) = OMEGL(I,J)*SIGT(KKLM)
99   CAPLL(I,J) = CAPLL(I,J)*SIGC(KKLM)
13   GO TO 13
11   WRITE(6,210)
11   GO TO 14

12 WRITE(6,211)
13 WRITE(6,212)DELOAD,EPS,LSMAX,MAXM,IIRMAX,LCHMAX
  WRITE(6,213)NU,TKN,CHAR,SIGO,ELAST,S
  IF(NTHMAX.EQ.0) GO TO 17
  WRITE(6,216)
  WRITE(6,217)(TH(NTH),NTH=1,NTHMAX)
17 LAM=TKN/CHAR
  SOE=SIGO/ELAST
  OSE=• 5*SOE
  D1=1• -NU
  S1=1• +NU
  LAM2=LAM**2

```

```

CALL GEOM(IGEOM,CHAR)
CALL STIF(ISTIF,TKN,ELAST,CHAR)
CALL PTLOAD(ILLOAD,TKN,CHAR,SIGO)
S=S/CHAR
DEL=S/AK
DELSQ=DEL**2
TDLI=.5/DEL
TDEL =2.0*DEL
CALL ACOEFF
MNINIT=1
MNMAXO=MNMAX
DO 20 I=1,4
DO 20 J=1,4
20 UNIT(I,J)=C.
DO 21 I=1,4
DO 21 J=1,4
21 UNIT(I,J)=1.
CALL PMATRX
DO 22 MN=1,MAXM
DO 22 K=1,KMAX2
DO 22 I=1,4
22 Z(I,K,MN)=C.
DO 230 M=1,MAXM
MAXD(M)=C
MAXS(M)=C
MAXSY(M)=C
230 ALLOAD=DELOAD
LSTEP=1
LCHANG=C
ITR=1
ICORFL=0
IPASS=0
ICTEST=0
400 CALL XANDZ
MNMAXO=MNMAX
IF(IPASS.LT.2) CALL MODES
IF(NCONV.EQ.1) GO TO 50
IF(ITR.LT.ITRMAX) GO TO 23
IF(LCHANG.LT.LCHMAX) GO TO 30

```

```

      WRITE(6,220) NO
      GO TO 500
      CALL OUTPUT(TKN,CHAR,SIGO)
      IF(LSTEP.EQ.LESMAX) GO TO 360
      IF(IPASS.EQ.1) 370,351,351
      370 IF(MNMAX.EQ.1) GO TO 351
      350 DO 355 MN=1,MNMAXO
      DO 355 K=1,KMAX2
      DO 355 I=1,4
      355 Z(I,K,MN)=2.*Z(I,K,MN)

      GO TO 62
      WRITE(6,221) NO
      GO TO 500
      351 ICTEST=ICTEST+1
      IF( ICTEST.GT.1) GO TO 60
      DO 352 MN=1,MNMAXO
      DO 352 K=1,KMAX2
      DO 352 I=1,4
      352 ZO(I,K,MN)=Z(I,K,MN)
      GO TO 350
      60 DO 61 MN=1,MNMAXO
      DO 61 K=1,KMAX2
      DO 61 I=1,4
      ZN=2.*Z(I,K,MN)-ZO(I,K,MN)
      ZO(I,K,MN)=Z(I,K,MN)
      61 Z(I,K,MN)=ZN
      62 ALLOAD=ALLOAD+DELOAD
      LSTEP=LSTEP+1
      ITR=1
      GO TO 400
      23 ITR=ITR+1
      GO TO 400
      34 IF(LSTEP.EQ.1) 310,310,320
      314 WRITE(6,223)

```

```

GO TO 500
32U WRITE(6,222)
LCHANG=LCHANG+1
LSTEP=LSTEP-1
ALOAD=ALOAD-DELOAD
DELOAD=DELOAD/5•
IF(ICTEST•GE•1) GO TO 325
DO 330 MN=1,MNMAXO
DO 330 K=1,KMAX2
DO 330 I=1,4
33U Z(I,K,MN)=.5*Z(I,K,MN)
GO TO 62
325 DO 31 MN=1,MNMAXO
DO 31 K=1,KMAX2
DO 31 I=1,4
31 Z(I,K,MN)=ZC(I,K,MN)
GO TO 62
100 FORMAT(12A6)
101 FORMAT(14I5)
102 FORMAT(6E12•3)
103 FORMAT(4E16•8)
105 FORMAT(6E12•3)
201 FORMAT(1H1,16H PROBLEM NUMBER 16//)
202 FORMAT(1X12A6//)
203 FORMAT(50X1H INPUT DATA/20H BOUNDARY CONDITIONS//)
204 FORMAT(2X,12HINITIAL EDGE,6X,9HOME GA BAR,25X,3HE
1L1//)
205 FORMAT(1P4E12•4,5X,1P4E12•4,5X,1PE12•4)
210 FORMAT(13H INITIAL POLE//)
211 FORMAT(11H FINAL POLE//)
212 FORMAT(//17HLOAD INCREMENT =,F6•3,27X,23HCONVERGENCE CRITERION =
1 F7•4//31H MAXIMUM NUMBER OF LOAD STEPS =,I3,16X25HMAXIMUM
20F MODES = I3//31H MAXIMUM NUMBER OF ITERATIONS =,I3•16X,42HMAXIMU
3M NUMBER OF LOAD INCREMENT CHANGES =,I2//)
213 FORMAT(17H POISONS RATIO =E16•5,17X, 21HREFERENCE THICKNESS =
1E13•5// 33H CHARACTERISTIC SHELL DIMENSION =E13•5,4X, 18HREFERENCE
2 STRESS = E13•5// 23H REFERENCE ELASTICITY =E13•5,14X,18HMERIDIAN

```

```
3 LENGTH = ,E13•5)
216 FORMAT(1H0,6H THETA)
217 FORMAT(1H ,5E16•8)
220 FORMAT(1H1,20H END PROBLEM NUMBER 16/14H LCHANG=LCHMAX)
221 FORMAT(1H1,20H END PROBLEM NUMBER 16/12H LSTEP=LSMAX)
222 FORMAT(1H1,39H CONVERGENCE FAILURE, STEP SIZE REDUCED)
223 FORMAT(1H1,45H CONVERGENCE FAILURE ON FIRST LOAD INCREMENT•)
224 FORMAT(2X,10HFINAL EDGE,8X,9HOMEGA BAR,43X,10HLAMBDA BAR,25X,3HELL
1 //)
END
```

```

SUBROUTINE GEOM( IGEOM,CHAR)
C   SUBROUTINE GEOM--THIS SUBROUTINE READS THE RADIUS AT EACH STATION
C   AND CALCULATES GAM,OMXI,OMT AND DEOMX BY DIFFERENCE APPROXIMATIONS.
C   THREE POINT CENTRAL DIFFERENCES ARE USED FOR INTERIOR POINTS, AND
C   THREE POINT FORWARD DIFFERENCES ARE USED FOR END POINTS. SPECIAL
C   FORMULAS ARE USED FOR POLES.
COMMON /IBL5/IBCINL,IBCFNL,IBL4/KMAX,KL/BL17/DEL/BL12/TDLI,TDEL/
1BL8/R(20),GAM(20),OMT(20)/BL20/DEOMX(20)/IBL6/KLL/BL31/DELSQ
2/BL11/OMXI(20),PHEE,T0,T2
FDIFF(A,B,C)=(-1.5*A+2.*B-.5*C)/DEL

C
C      READ (5,101) (R(K),K=1,20)

C
C      IF(IGEOM) 500, 50, 50

C
C      50 DO 10 K = 2,KMAX
      RK = K
      THET = (RK-1.0)*DEL/1000.
      R(K) = SIN(THET) * 1000.
      GAM(K)=COS(THET)/R(K)
      OMT(K)=1.0/1000.
      OMXI(K)=1.0/1000.
      DEOMX(K)=0.0
      R(1)=0.0
      GAM(1)=0.0
      OMT(1) = 1.0/1000.
      OMXI(1) = 1.0/1000.
      DEOMX(1) = 0.0
      GO TO 1000
      10

```

```

C      500   RD2 = 1.0/DELSQ
      DO 1 K=2,KL
      DR=(-R(K-1)+R(K+1))*TDLI
      GAM(K)=DR/R(K)
      ROOT=SQRT(1.-DR**2)
      OMT(K)=ROOT/R(K)
      1  OMXI(K)=-(R(K-1)-2.*R(K)+R(K+1))*RD2/ROOT
      IF(IBCINL.LT.0) GO TO 2
      DR=FDIFF(R(1),R(2),R(3))
      GAM(1)=DR/R(1)
      ROOT=SQRT(1.-DR**2)
      OMT(1)=ROOT/R(1)
      OMXI(1)=-(2.*R(1)-5.*R(2)+4.*R(3)-R(4))*RD2/ROOT
      DEOMX(1)=FDIFF(OMXI(1),OMXI(2),OMXI(3))
      GO TO 3
      C      GAMA IS INFINITE AT A POLE BUT IS SET TO ZERO FOR DEFINITNESS
      2  GAM(1)=0.
      OMT(1)=SQRT((2.*R(2)-R(3))/DEL**3)
      OMXI(1)=OMT(1)
      DEOMX(1)=0.
      3  IF(IBCINL.LT.0) GO TO 4
      DR=-FDIFF(R(KMAX),R(KL),R(KLL))
      GAM(KMAX)=DR/R(KMAX)
      ROOT=SQRT(1.-DR**2)
      OMT(KMAX)=ROOT/R(KMAX)

```

```

OMXI(KMAX)=- (2.*R(KMAX)-5.*R(KL)+4.*R(KLL)-R(KLL-1))*RD2/ROOT
DEOMX(KMAX)=-FDIFF(OMXI(KMAX),OMXI(KL),OMXI(KLL))
GO TO 5
4 GAM(KMAX)=0.
C   GAMA IS INFINITE AT A POLE BUT IS SET TO ZERO FOR DEFINITNESS
OMT(KMAX)=SQRT((-2.*R(KL)-R(KLL))/DEL**3)
OMXI(KMAX)=OMT(KMAX)
DEOMX(KMAX)=0.
5 DO 6 K=2,KL
6 DEOMX(K)=(-OMXI(K-1)+OMXI(K+1))*TDL1
101 FORMAT(7E10.1)
1000 WRITE(6,102)
102 FORMAT(1H1.9H STATION 14H RADIUS 14H GAMMA 13H OMEG
1A S 15H OMEGA THETA13H DEOMEGA S //)
WRITE(6,103)(K,R(K),GAM(K),OMX1(K),OMT(K),DEOMX(K),K=1,KMAX)
103 FORMAT(14.5X,5E14.4)
CHAR2=CHAR*CHAR
DO 979 K=1,KMAX
R(K)=R(K)/CHAR
OMXI(K)=OMXI(K)*CHAR
GAM(K)=GAM(K)*CHAR
OMT(K)=OMT(K)*CHAR
979 DEOMX(K)=DEOMX(K)*CHAR2
RETURN
END

```

```

C      SUBROUTINE STIF(ISTIF,TKN,ELAST,CHAR)
C      SUBROUTINE STIF--READS B AND D AT EACH STATION AND CALCULATES
C      DB AND DD. USING THREE POINT CENTRAL DIFFERENCES FOR INTERIOR
C      POINTS AND THREE POINT FORWARD DIFFERENCES AT END POINTS.
C      COMMON/IBL4/KMAX,KL/IBL6/KLL/BL2/BL17/DEL/BL12/TDL1,TDEL
C      120)/BL12/TDL1,TDEL
COMMON /BL15/NU,U1(10),V1(10),W1(10),U2(10),V2(10),W2(10),U3(10),
1 V3(10),W3(10)

1      REAL NU
FDIFF(A,B,C)=(-1.5*A+2.*B-.5*C)/DEL

C
C
C      READ (5,101) (B(K), K=1,20)
READ (5,101) (D(K), K=1,20)

C
C      ZN = ELAST*TKN*(1. -NU**2)
ZN = ZN*TKN*TKN
IF (ISTIF) 500, 50, 50

C
C      DO 9 K = 1, KMAX
50   B(K) = 27.3 E6
DB(K) = 0.0
DD(K) = 0.0
9  D(K) = 2.275 E6
GO TO 1000

```

```

C      DB(1) = FDIFF(B(1)*B(2)*B(3))
C      DD(1)=FDIFF(D(1)*D(2),D(3))
DB(KMAX)=-FDIFF(B(KMAX),B(KL)*B(KLL))
DD(KMAX)=-FDIFF(D(KMAX),D(KL)*D(KLL))
DO 1 K=2,KL
DB(K)=(-B(K-1)+B(K+1))*TDLI
1 DD(K)=(-D(K-1)+D(K+1))*TDLI
101 FORMAT(7E10.1)
1000 WRITE(6,102)
102 FORMAT(1H1,9HSTATION   14H    B
          14H    D
          14H    DB
          14H    DD
          14H    //)
          WRITE(6,103)(K,B(K),D(K),DB(K),DD(K),K=1,KMAX)
103 FORMAT(14,5X,4E14.4)
DO 979 K=1,KMAX
B(K)=B(K)/ZN
DB(K)=DB(K)*CHAR/ZN
D(K)=D(K)/ZNN
979 DD(K)=DD(K)*CHAR/ZNN
RETURN
END

```

```

C SUBROUTINE PTLOAD(ILOAD,TKN,CHAR,SIGO)
C SUBROUTINE PTLOAD--READS THE LOAD QUANTITIES PR,PX,PT,TT AND MT AT
C EACH STATION FOR EACH MODE. ALSO READS THE INITIAL VALUE OF
C MNMAX, READS THE N FOR EACH MODE, AND SETS UP THE
C N(MN) ARRAY.
C
REAL NU
REAL MT
COMMON /IBL1/MNMAX/IBL4/KMAX,KL/IBL6/KLL/IBL2/N(10),MNINT
1 /IBL3/MC,M1,M2,
1M3/BL17/DEL/BL12/TDEL,TDEL
1 /BL3/PR(20,10),PX(20,10),PT(20,10)/BL5/TT(20
2,10),MT(20,10),DTT(20,10),DMT(20,10)/IBL9/MAXM
COMMON /BL15/NU,U1(10),V1(10),W1(10),U2(10),V2(10),W2(10),
1 U3(10),V3(10),W3(10)
FDIFF(A,B,C)=(-1.5*A+2.*B-.5*C)/DEL
MU=0.
M1=0.
M2=0.
M3=0.
DO 917 MN=1,10
DO 917 K=1,20
PR(K,MN)=0.
PX(K,MN)=0.
PT(K,MN)=0.
TT(K,MN)=0.
MT(K,MN)=0.
DTT(K,MN)=0.
DMT(K,MN)=0.
917
C
C
DO 1 MN = 1,MNMAX
READ (5,102) N(MN)
READ (5,101) (PR(K,MN), K =1,20)
READ (5,101) (PX(K,MN), K =1,20)
READ (5,101) (PT(K,MN), K =1,20)

```

```

1 READ (5,101) (TT(K,MN), K =1,20)
  READ (5,101) (MT(K,MN), K =1,20)
C C
C ABN = CHAR/(SIGO*TKN)
ZN=SIGO*TKN*(1.-NU)
IF (ILOAD) 500, 50, 50
C
C 50 DO 989 MN = 1, MNMAX
DO 989 K = 1, KMAX
N(1)=0.0
N(2) = 1
N(3) = 3
PR(K,1) = -15.0
PR(K,2) = -19.1
PR(K,3) = 6.37
PX(K,MN) = 0.0
PT(K,MN) = 0.0
DTT(K,MN) = 0.0
DMT(K,MN) = 0.0
TT(K,MN) = 0.0
MT(K,MN) = 0.0
989 GO TO 1000
C
C 101 FORMAT(7E10.1)
102 FORMAT(14)
500 CONTINUE
DO 3 MN=1,MNMAX
DTT(1,MN)=FDIFF(TT(1,MN),TT(2,MN),TT(3,MN))
DTT(KMAX,MN)=-FDIFF(TT(KMAX,MN),TT(KL,MN),TT(KLL,MN).)
DMT(1,MN)=FDIFF(MT(1,MN),MT(2,MN),MT(3,MN))
DMT(KMAX,MN)=-FDIFF(MT(KMAX,MN),MT(KL,MN),MT(KLL,MN))
DO 3 K=2,KL
DTT(K,MN)=TDLI*(-TT(K-1,MN)+TT(K+1,MN))
3 DMT(K,MN)=TDLI*(-MT(K-1,MN)+MT(K+1,MN))
1000 WRITE (6,1C3)

```

```

103 FORMAT(1H1,9HSTATION 7H N      14H P      14H PS
14H PT   14H TT    14H MT    14H DTT
2 14H DMT   //
      WRITE(6,104)((K,N(MN),PR(K,MN),PX(K,MN),PT(K,MN),TT(K,MN),
1      MT(K,MN),DTT(K,MN),DMT(K,MN),K=1,KMAX),MN=1,MNMAX)
104 FORMAT(14,5X,I3,4X,7E14.4)
DO 979 MN = 1,MNMAX
DO 979 K = 1,KMAX
      PR(K,MN) = PR(K,MN)*ABN
      PT(K,MN) = PT(K,MN)*ABN
      PX(K,MN) = PX(K,MN)*ABN
      TT(K,MN) = TT(K,MN)/ZN
      DTT(K,MN) = DTT(K,MN)*CHAR/ZN
      DMT(K,MN) = DMT(K,MN)*CHAR**2/(ZN*TKN**2)
      MT(K,MN) = MT(K,MN)*CHAR/(ZN*TKN**2)
979 RETURN
END

```

```

C SUBROUTINE ACOEFF--THIS SUBROUTINE CALCULATES THE QUANTITIES
C REQUIRED TO EXPRESS BUDJANSKY'S COEFFICIENTS AS POLYNOMIALS IN N.
REAL LAM2,LSD,LSDD,LSD1N,LSD18,NU
COMMON /IBL5/IBCINL,IBCFNL/IBL4/KMAX,KL/BL15/NU/BL14/LAM2,LSD18,L
1 LSD1N/BL7/D1,S1/BL8/R(20),GAM(20),OMT(20)/BL11/OMXI(20)/BL20/DEOMX
2 (20)/BL2/B(20),D(20),DB(20),DD(20)/BL21/A1NO(20),A2NO(20),A3NO(20
3 )•A3N2(20)•A4N1(20)•A5N1(20)•A6NO(20)•A7NO(20)•A7N2(20)•
4 A8NO(20)•A9NO(20)•A10N1(20)•A11N1(20)•A12NO(20)•A13NO(20)•A14NO(2
5 0)•A14N2(20)•A15N1(20)•A16N1(20)•A17N1(20)•A17N3(20)•A18N1(20)•A1
6 9NO(20)•A19N2(20)•A20NO(20)•A20N2(20)•A21N1(20)•A22N1(20)•A23N1(
7 20)•A23N3(20)•A24NO(20)•A24N2(20)•A25NO(20)•A25N2(20)•A26NO(20),
8 A26N2(20)•A26N4(20)•A27NO(20)•A28NO(20)•A29NO(20)•A29N2(20)•A30NO(
9 20)•A31NO(20)•A32N1(20)•A33NO(20)•A34NO(20)•A35N2(20)•A36NO(20)
D18=D1/8.
D12=D1/2.

KS=1
IF(IBCINL.LT.0)KS=2
KSTP=KMAX
IF( IBCFNL.LT.0) KSTP=KL
DO 1 K=KS,KSTP
RA=R(K)
GA=GAM(K)
OX=OMXI(K)
OT=OMT(K)
DEX=DEOMX(K)
GA2=GA**2
REX=(3.*OT-OX)
RXE=(3.*OX-OT)
OTX=OT*OX
BS=B(K)
DS=D(K)
DBS=DB(K)
DDS=DD(K)
LSD=LAM2*DS
RRA=1./RA
LSDD=LAM2*DDS
RRA2=RRA**2

```

```

LSD1N=LSD*D1
LSD18=LSD*D18
A1NO(K)=BS
A2NO(K)=GA*BS+DBS
A3NO(K)=NU*DBS*GA-NU*BS*OTX-BS*GA2-LSD1N *S1*GA2*OX***2
A3N2(K)=-D12 *BS*RR2-LSD18 *RRA2*RXE***2
A4N1(K)=(•5*S1*BS+LSD18 *RXE*REX)*RRA
A5N1(K)=(NU*DBS-(3.-NU)*.5*GA*BS-LSD1N*GA*(•125*REX*RXE+S1*
10TX)*RRA
A6NO(K)=BS*(OX+NU*OT)+LSD1N*S1*GA2*OX
A6N2(K)=LSD1N*.5*RXE*RR2
A7NO(K)=BS*(DEX+GA*(OX-OT))+DBS*(OX+NU*OT)
A7N2(K)=-LSD1N*GA*RR2*(•5*RXE+S1*OX)
A8NC(K)=LAM2*OX
A9NO(K)=LAM2*D1*GA*OX
A10N1(K)=-A4N1(K)
A11N1(K)=(-.5*BS*GA*(3.-NU)-D12*DBS+LSD1N*(-S1*GA*OTX+.125*GA*
1(6.*OTX-7.*OX**2-3.*OT**2)-.25*DEX*(5.*OT-3.*OX))-LSDD*D18*RXE
2*REX)*RRA
A12NO(K)=.5*BS*D1+LSD18*REX**2
A13NO(K)=D12*A2NO(K)-LSD18*REX*(2.*DEX-GA*(5.*OX-3.*OT))+LSDD
1*.125*D1*REX**2
A14NO(K)=-GA*A13NO(K)+D12*BS*OTX+LSD18*OTX*REX**2

A14N2(K)=-BS*RR2-LSD1N*S1*RR2*OT***2
A15N1(K)=LSD1N*.5*REX*RR2
A16N1(K)=(LSD1N*.5*(2.*S1*GA*OT-DEX+3.*GA*(OX-OT))+LSDD*D12
1*REX)*RRA
A17N1(K)=(-BS*(OT+NU*OX)+LSD*D12*(GA*DEX-2.*GA2*OX+REX*(GA2+OTX))
1-LSDD*D12*GA*REX)*RRA
A17N3(K)=-LSD1N*S1*OT*RR2***3
A18N1(K)=-NU*LAM2*OT*RR2
A19NO(K)=-A6NO(K)
A19N2(K)=-A6N2(K)
A20NO(K)=-BS*GA*(-GA*DEX+GA*S1*(-GA*DEX+GA2*OX+2.*OT*

```

```

10X**2)-LSDD*D1*S1*GA2*OX
A20N2(K)=LSD1N*(GA*S1*(-OX*RRA2)+.5*RRA2*(GA*(OX-OT)-3.*DEX))
1-LSDD*D12*RXE*RRA2
A21N1(K)=A15N1(K)
A22N1(K)=(LSD*D12*(3.*GA*OX-GA*OT*(5.+2.*NU)-DEX)+LSDD*D12*
1REX)*RRA
A23N1(K)=(-BS*(OT+NU*OX)+LSD*D12*(2.*S1*(OTX*OT-GA2*OX+2.*GA2
1*OT)+GA*DEX+3.*GA2*(OT-OX)+OTX*REX)-LSDD*D12*(2.*S1*GA*OT+GA
2*REX))*RRA
A23N3(K)=-LSD1N*S1*OT*RRA**3
A24N0(K)=LSD1N*S1*GA2
A24N2(K)=2.*LSD1N*RRA2
A25N0(K)=-LSD1N*S1*(2.*GA*OTX+GA2*GA)+LSDD*D1*S1*GA2
A25N2(K)=-LSD1N*2.*GA*RRA2+LSDD*D1*2.*RRA2
A26N0(K)=-BS*(OX*OX+2.*NU*OTX+OT*OT)
A26N2(K)=(LSD1N*(S1*(OTX+2.*GA2)+2.*(GA2+OTX))-LSDD*D1*(3.+NU)
1*GA)*RRA2
A26N4(K)=-LSD1N*S1*RRA2**2
A27N0(K)=LAM2
A28N0(K)=LAM2*GA*(2.-NU)
A29N0(K)=-LAM2*D1*OTX
A29N2(K)=-LAM2*NU*RRA2
A30N0(K)=DS*OX
A31N0(K)=DS*(DEX+NU*GA*OX)
A32N1(K)=DS*NU*OT*RRA
A33N0(K)=-DS
A34N0(K)=-DS*NU*GA
A35N2(K)=DS*NU*RRA2
1 A36N0(K)=-1.
1 RETURN
END

```

```

SUBROUTINE PMATRIX
SUBROUTINE PMATRIX--CALCULATES P(K,MN) AND OTHER MATRICES
C
C INDEPENDENT OF THE LOAD.
REAL JAY
COMMON /IBL1/MNMAX/IBL5/IBCINL,IBCFNL/IBL2/N(10),MNINIT/IBL4/KMAX,
1KL/IBL3/M0,M1,M2,M3/BL13/OMEG1(4,4),CAPL1(4,4),OMEGL(4,4),CAPLL(4
2,4),UNIT(4,4)/BL1/A(4,4),BEE(4,4),C(4,4)
3/BL23/JAY(4,4),H(4,4)/BL24/DL(4,4,10),
4DG(4,4,10),DF(4,4,10)/BL4/P(16,20,10),X(4,20,10),ZFLM(4,4,10),ZF
52M(4,4,10),ZF3M(4,4,10),ZF4M(4,4,10)
DIMENSION PATA(4,4),PBTA(4,4),POTA(4,4),PJTA(4,4),DLL(4,4),PTR(4,
14),DGG(4,4),ZFL1(4,4),ZF2(4,4),ZFP0(4,4),ZFP1(4,4),ZFP2(4,4),IPIVO
2T(4),INDEX(4,2),CL0(4,4),CL1(4,4),CL2(4,4)
EQUIVALENCE (CL0(1),ZF1M(1)),(CL1(1),ZF2M(1)),(CL2(1),ZF3M(1)),
1(ZFPC(1),PATA(1)),(ZFP1(1),PBTA(1)),(ZFP2(1),POTA(1))
2,(ZF1(1),DLL(1)),(ZF2(1),PTR(1))
IF(IBCINL.LT.0) GO TO 10
DO 1 MN=MNINIT,MNMAX
CALL HJ(1,MN)
CALL EFG(1,MN)
CALL ABC
CALL MATINV(C,4,G1,0,DETERM,IPIVOT,INDEX,4,ISCALE)
DO 3 J=1,4
DO 3 I=1,4
SUMA=0.
SUMB=0.
SUMO=0.
SUMJ=0.
DO 4 L=1,4
SUMJ=SUMJ+OMEG1(I,L)*JAY(L,J)
SUMA=SUMA+C(I,L)*A(L,J)
SUMB=SUMB+C(I,L)*BEE(L,J)
4 SUMO=SUMO+OMEG1(I,L)*H(L,J)
PATA(I,J)=SUMA+UNIT(I,J)
PBTA(I,J)=SUMB
POTA(I,J)=SUMO
3 PJTA(I,J)=SUMJ
DO 5 J=1,4
DO 5 I=1,4

```

```

SUMOB=0.
SUMOC=0.
SUMOA=0.

DO 6 L=1,4
SUMOB=SUMOB+POTA(I,L)*PBTA(L,J)
SUMOA=SUMOA+POTA(I,L)*PATA(L,J)
6 SUMOC=SUMOC+POTA(I,L)*C(L,J)
DLL(I,J)=SUMOB+PJTA(I,J)+CAPL1(I,J)
PTR(I,J)=SUMOA

5 DGG(I,J)=SUMOC
CALL MATINV(DLL,4,PTR,4,DETERM,IPIVOT,INDEX,4,ISCALE)

DO 1 J=1,4
DO 1 I=1,4
SUMD=0.
SUME=0.
DO 7 L=1,4
SUMD=SUMD+DLL(I,L)*DGG(L,J)
7 SUME=SUME-DLL(I,L)*OMEG1(L,J)
DL(I,J,MN)=DLL(I,J)
DG(I,J,MN)=SUMD
DF(I,J,MN)=SUME
IJ=I+4*(J-1)

1 P(IJ,1,MN)=PTR(I,J)
GO TO 20
10 DO 11 MN=MNINIT,MNMAX
NN=N(MN)
DO 14 I=1,16
14 P(I,1,MN)=0.
DO 15 I=1,4
15 X(I,1,MN)=0.
IF(NN.GT.3) GO TO 11
IF(NN.GT.2) GO TO 90
IF(NN.GT.1) GO TO 12
IF(NN.GT.0) GO TO 13

```

```

P(11,1,MN)=-1.
P(16,1,MN)=-1.
M0=MN
GO TO 11
12 P(16,1,MN)=-1.
M2=MN
GO TO 11
90 M3=MN
GO TO 11
13 P(1,1,MN)=-1.
P(2,1,MN)=1.
M1=MN
11 CONTINUE
20 KLAST=KMAX
IF(IBCFLN.LT.0) KLAST=KL
DO 23 K=2,KLAST
DO 23 MN=MNINIT,MNMAX
CALL EFG(K,MN)
CALL ABC
23 CALL PANDD(K,MN)
IF(IBCFLN.LT.0) GO TO 30
DO 40 MN=MNINIT,MNMAX
CALL HJ(KMAX,MN)
DO 41 J=1,4
DO 41 I=1,4
SUMO=0.
SUMP=0.
SUMJ=0.
DO 42 L=1,4
SUMO=SUMO+OMEGL(I,L)*H(L,J)
IL=I+4*(L-1)
LJ=L+4*(J-1)
SUMP=SUMP+P(IL,KL,MN)*P(LJ,KMAX,MN)
42 SUMJ=SUMJ+OMEGL(I,L)*JAY(L,J)
PATA(I,J)=SUMO
PBTA(I,J)=UNIT(I,J)-SUMP
41 PJTA(I,J)=SUMJ+CAPLL(I,J)
DO 43 J=1,4
DO 43 I=1,4

```

```

SUMOP=0.
SUMJP=0.
SUMOM=0.
DO 44 L=1,4
LJ=L+4*(J-1)
SUMOP=SUMOP+PATA(I,L)*PBTA(L,J)
SUMJP=SUMJP+PJTA(I,L)*P(LJ,KMAX,MN)
SUMOM=SUMOM-PATA(I,L)*P(LJ,KL,MN)
ZF1(I,J)=SUMOP-SUMJP
43 ZF2(I,J)=SUMOM-PJTA(I,J)

```

```

CALL MATINV(ZF1,4,ZF2,4,DETERM,IPIVOT,INDEX,4,ISCALE)
DO 45 J=1,4
DO 45 I=1,4
SZF3=0.
SZF4=0.
DO 46 L=1,4
SZF3=SZF3+ZF1(I,L)*PATA(L,J)
46 SZF4=SZF4-ZF1(I,L)*OMEGL(L,J)
ZF3M(I,J,MN)=SZF3
ZF4M(I,J,MN)=SZF4
ZF1M(I,J,MN)=ZF1(I,J)
ZF2M(I,J,MN)=ZF2(I,J)
45 CONTINUE
40 RETURN
30 K1 = KL
DO 31 MN = MNINIT,MNMAX
NN=N(MN)
IF(NN .GT. 3) GO TO 31
IF(NN.GT.2) GO TO 300
IF(NN .GT. 1) GO TO 33
IF(NN .GT. 0) GO TO 34
MO=MN
DO 35 J=1,4
DO 35 I=1,4

```

```

      CL0(I,J)=0.
  35  ZFP0(I,J)=0.
      ZFP0(1,1)=1.
      ZFP0(2,2)=1.
      ZFP0(3,1)=P(3,KL,MN)
      ZFP0(3,2)=P(7,KL,MN)
      ZFP0(3,3)=P(11,KL,MN)+1.
      ZFP0(3,4)=P(15,KL,MN)
      ZFP0(4,1)=P(4,KL,MN)
      ZFP0(4,2)=P(8,KL,MN)
      ZFP0(4,3)=P(12,KL,MN)
      ZFP0(4,4)=P(16,KL,MN)+1.
      CL0(3,3)=-1.
      CL0(4,4)=-1.
      CALL MATINV(ZFP0,4,CL0,4,DETERM,IPIVOT,INDEX,4,ISCALE)
      GO TO 31
  300 M3=MN
      GO TO 31
  34 M1=MN
      DO 60 J=1,4
      DO 60 I=1,4
      CL1(I,J)=0.
  60 ZFP1(I,J)=0.
      ZFP1(1,1)=P(1,K1,MN)+1.
      ZFP1(1,2)=P(5,K1,MN)
      ZFP1(1,3)=P(9,K1,MN)
      ZFP1(1,4)=P(13,K1,MN)
      ZFP1(2,1)=1.
      ZFP1(2,2)=1.
      ZFP1(3,3)=1.
      ZFP1(4,4)=1.
      CL1(1,1)=-1.
      CALL MATINV(ZFP1,4,CL1,4,DETERM,IPIVOT,INDEX,4,ISCALE)
      GO TO 31
  33 M2=MN
      DO 70 J=1,4
      DO 70 I=1,4

```

```
CL2(I,J)=0.  
70 ZFP2(I,J)=0.  
ZFP2(I,1)=1.  
ZFP2(2,2)=1.  
ZFP2(3,3)=1.  
ZFP2(4,1)=P(4,K1,MN)  
ZFP2(4,2)=P(8,K1,MN)  
ZFP2(4,3)=P(12,K1,MN)  
ZFP2(4,4)=P(16,K1,MN)+1.  
CL2(4,4)=-1.  
CALL MATINV(ZFP2,4,CL2,4,DETERM,IPIVOT,INDEX,4,ISCALE)  
31 CONTINUE  
RETURN  
END
```

```

C      SUBROUTINE HJ(K,MN)
C      SUBROUTINE HJ--THIS SUBROUTINE CALCULATES THE H AND JAY MATRICES
C      FOR GIVEN K AND MN. NOTE H HAS BEEN DIVIDED BY 2.*DEL.
REAL LAM2,LSD1N,LSD12,LSD18,JAY,NU
COMMON /IBL4/KMAX/IBL2/N(10),MNNINIT/BL14/LAM2,LSD18,NU/BL12/TD
1LI/BL7/D1,S1/BL23/JAY(4,4),H(4,4)/BL15/NU/BL8/R(20),GAM(20),OMT
2(20)/BL2/B(20),D(20)/BL11/OMX1(20)
DIMENSION B1NO(2),B2NO(2),B4NO(2),B6NO(2),B7NO(2),B10NO(2),B13NO(
12),B15NO(2),B16NO(2),B3N1(2),B5N1(2),B8N1(2),B9N1(2),B11N1(2),B12
2N1(2),B10N2(2),B13N2(2),B14N2(2)

L=1
IF(K.EQ.KMAX) L=2
C      THE BRANCH BELOW AVOIDS RECALCULATION OF ELEMENTS WHICH ARE
C      CONSTANT WITH RESPECT TO N.
IF(MN.GT.MNNINIT) GO TO 1
GA=GAM(K)
OX=OMX1(K)
OT=OMT(K)
OXT=3.*OX-OT
OTX=3.*OT-OX
BS=B(K)
DS=D(K)
LSD12=LSD1N*.5
GA2=GA**2
RRA=1./R(K)
RRA2=RRA**2
B1NO(L)=BS
B2NO(L)=NU*GA*BS
B4NO(L)=BS*(OX+NU*OT)
B6NO(L)=.5*BS*D1+LSD18*OTX**2
B7NO(L)=-GA*B6NO(L)
B10NO(L)=-LSD1N*S1*OX*GA2
B13NO(L)=LSD1N*S1*GA2
B15NO(L)=LAM2
B16NO(L)=LAM2*D1*GA
B3N1(L)=NU*BS*RRA
B5N1(L)=-(.5*BS*D1+LSD18*OTX*OTX)*RRA
B8N1(L)=LSD12*OTX*RRA

```

```

B9N1(L)=-GA*B8N1(L)
B11N1(L)=B8N1(L)
B12N1(L)=-LSD12*GA*(OTX+2.*S1*OT)*RRA
B10N2(L)=-LSD12*OXT*RRA2
B13N2(L)=2.*LSD1N*RRA2
B14N2(L)=-LSD1N*(3.+NU)*GA*RRA2
H(1,1)=B1NO(L)*TDLI
H(1,2)=0.
H(1,3)=0.
H(1,4)=0.
H(2,1)=0.
H(2,2)=B6NO(L)*TDLI
H(2,4)=0.
H(3,1)=0.
H(3,4)=B15NO(L)*TDLI
H(4,1)=0.
H(4,2)=0.
H(4,3)=-1.*TDLI
H(4,4)=0.
JAY(1,1)=B2NO(L)
JAY(1,3)=B4NO(L)
JAY(1,4)=0.
JAY(2,2)=B7NO(L)

```

```

JAY(2,4)=0
JAY(3,4)=B16NO(L)
JAY(4,1)=OX
JAY(4,2)=0
JAY(4,3)=0
JAY(4,4)=0.
EN=N(MN)
EN2=EN**2
H(2,3)=B8N1(L)*EN*TDLI
H(3,2)=B11N1(L)*EN*TDLI
H(3,3)=(B13NO(L)+B13N2(L)*EN2)*TDLI
JAY(1,2)=B3N1(L)*EN
JAY(2,1)=B5N1(L)*EN
JAY(2,3)=B9N1(L)*EN
JAY(3,1)=B10NO(L)+B10N2(L)*EN2
JAY(3,2)=B12N1(L)*EN
JAY(3,3)=B14N2(L)*EN2
RETURN
END

```

```

C SUBROUTINE EFG(K,MN) SUBROUTINE CALCULATES THE E,F AND G MATRICES
C FOR GIVEN K AND MN, USING THE QUANTITIES GENERATED BY SUBROUTINE
C ACOEFF.
COMMON /IBL2/N(10),MNINIT/BL21/A1NO(20),A2NO(20),A3NO(20),
1)•A4N1(20),A5N1(20),A6NU(20),A6N2(20),A7NO(20),A7N2(20),
2A9NO(20),A10N1(20),A11N1(20),A12NO(20),A13NO(20),A14N2(
320),A15N1(20),A16N1(20),A17N1(20),A17N3(20),A18N1(20),A
419N2(20),A20NO(20),A20N2(20),A21N1(20),A22N1(20),A23N3
5(20)•A24NO(20),A24N2(20),A25NO(20),A25N2(20),A26NO(20),A26N2(20),
6A26N4(20),A27NO(20),A28NO(20),A29NO(20),A29N2(20),A30NO(20),A31NO
7(20),A32N1(20),A33NO(20),A34NO(20),A35N2(20),A36NO(20)/BL25/E(4,4
8),F(4,4),G(4,4)
EN=N(MN)
EN2=EN**2
EN3=EN2*EN
EN4=EN3*EN
IF(MN.GT.MNINIT) GO TO 1
C THIS BRANCH AVOIDS RECALCULATION OF ELEMENTS WHICH ARE CONSTANT
C WITH RESPECT TO N.
E(1,1)=A1NO(K)
E(1,2)=0.
E(1,3)=0.
E(1,4)=0.
E(2,1)=0.
E(2,2)=A12NO(K)
E(2,4)=0.
E(3,1)=0.
E(3,4)=A27NO(K)
E(4,1)=0.
E(4,2)=0.
E(4,3)=A33NO(K)
E(4,4)=0.
F(1,1)=A2NO(K)
F(1,4)=A8NO(K)
F(2,2)=A13NO(K)
F(2,4)=0.
F(3,4)=A28NO(K)
F(4,1)=A30NO(K)

```

```

F(4,2)=0.
F(4,3)=A34NO(K)
F(4,4)=0.
G(1,4)=A9NO(K)
G(4,1)=A31NO(K)
G(4,4)=A36NO(K)
1 E(2,3)=A15N1(K)*EN
E(3,2)=A21N1(K)*EN
E(3,3)=A24NO(K)+A24N2(K)*EN2
F(1,2)=A4N1(K)*EN
F(1,3)=A6NO(K)+A6N2(K)*EN2
F(2,1)=A10N1(K)*EN
F(2,3)=A16N1(K)*EN
F(3,1)=A19NO(K)+A19N2(K)*EN2
F(3,2)=A22N1(K)*EN
F(3,3)=A25NO(K)+A25N2(K)*EN2
G(1,1)=A3NO(K)+A3N2(K)*EN2
G(1,2)=A5N1(K)*EN
G(1,3)=A7NO(K)+A7N2(K)*EN2
G(2,1)=A11N1(K)*EN
G(2,2)=A14NO(K)+A14N2(K)*EN2
G(2,3)=A17N1(K)*EN+A17N3(K)*EN3
G(2,4)=A18N1(K)*EN
G(3,1)=A20NO(K)+A20N2(K)*EN2
G(3,2)=A23N1(K)*EN+A23N3(K)*EN3
G(3,3)=A26NO(K)+A26N2(K)*EN2+A26N4(K)*EN4
G(3,4)=A29NO(K)+A29N2(K)*EN2
G(4,2)=A32N1(K)*EN
G(4,3)=A35N2(K)*EN2
RETURN
END

```

```

C          SUBROUTINE ABC--THIS SUBROUTINE CALCULATES THE A,BEE AND C
C          SUBROUTINE ABC--THIS SUBROUTINE CALCULATES THE A,BEE AND C
C          MATRICES AT ALL STATIONS FOR ALL MODES
COMMON /BL17/DEL/BL12/TDL1,TDL1/A(4,4),BEE(4,4),C(4,4)/BL25/
1E(4,4),F(4,4),G(4,4)
D2=2.*DEL
DO 1 J=1,4
DO 1 I=1,4
DEIJ=D2*E(I,J)
FIJ=F(I,J)
BEE(I,J)=-2.*DEIJ+TDEL*G(I,J)
C(I,J)=DEIJ-FIJ
1 A(I,J)=DEIJ+FIJ
      RETURN
END

```

```

SUBROUTINE PANDD(K,MN)
C   SUBROUTINE PANDD--THIS SUBROUTINE CALCULATES THE P,DEE AND DST
C   MATRICES FOR GIVEN K AND MN, USING THE MATRICES GENERATED BY
C   SUBROUTINE ABC.
COMMON /BL1/A(4,4),BEE(4,4),C(4,4)/BL4/P(16,20,10)
COMMON DEE(16,20,10),DST(16,20,10)
DIMENSION TM(4,4),IPIVOT(4),INDEX(4,2)
DO 1 I=1,4
DO 1 J=1,4
SUM=0.
DO 2 L=1,4
LJ=L+4*(J-1)
SUM=SUM+C(I,L)*P(LJ,K-1,MN)
2   TM(I,J)=BEE(I,J)-SUM
CALL MATINV(TM,4,X2,0,DETERM,IPIVOT,INDEX,4,ISCALE)
DO 5 I=1,4
DO 5 J=1,4
SUMA=0.
SUMC=0.
DO 6 L=1,4
SUMA=SUMA+TM(I,L)*A(L,J)
6   SUMC=SUMC+TM(I,L)*C(L,J)
IJ=I+4*(J-1)
P(IJ,K,MN)=SUMA
DEE(IJ,K,MN)=TM(I,J)
5   DST(IJ,K,MN)=SUMC
      RETURN
END

```

C
 SUBROUTINE XANDZ--CALCULATES THE X VECTOR AND SOLVES FOR THE
 Z VECTOR.
 C

REAL NU,LAM2,MT
 COMMON /IBL1/MNMAX,IBL5/IBCNL,IBCFNL/IBL4/KMAX,KL/IBL7/MNMAXO/
 1IBL12/KMAX1,KMAX2,NCONV/IBL6/KLL/IBL3/M0,M1,M2,M3/BL16/EPS/BL15/
 2NU,U1(10),V1(10),W1(10),U2(10),V2(10),W2(10),U3(10),V3(10),W3(10)
 3/BL18/EL1(4),ELL(4)/BL14/LAM2/BL6/Z(4,22,10)*SOE,OSE,ALOAD/BL12/
 4TDL1/BL7/D1/BL8/R(20),GAM(20)/BL2/B(20)/BL5/TT(20,10),MT(20,10)/
 5 BL4/P(16,20,10),X(4,20,
 610),ZF1M(4,4,10),ZF2M(4,4,10),ZF3M(4,4,10),ZF4M(4,4,10)
 7 /BL9/FFS(4,10),BE3(4),GEES(4,10)/BL27/BX3(10),
 8BT3(10),BXT3(10),BE3(10)/BL28/EXX3(10),ETT3(10),EXT3(10)
 9,EX3(10),ET3(10)
 COMMON /BL29/BX1(10),BT1(10),BX2(10),BE1(10),BX2(10),B
 1T2(10),BXT2(10),BE2(10)/BL30/EXX1(10),ETT1(10),ETX1(10),EXT1(10),
 2EX1(10),ET1(10),EXX2(10),ETT2(10),ETX2(10),EXT2(10),ET2(1
 30)/BL1/A(4,4),BEE(4,4),C(4,4)
 DIMENSION ELLS(4),FLS(4),ZT(4),IPIVOT(4),INDEX(4,2)
 1•CL0(4,4),CL1(4,4),CL2(4,4)
 EQUIVALENCE (CL0(1),ZF1M(1)),(CL1(1),ZF2M(1)),(CL2(1),ZF3M(1))
 NCONV=1
 DO 1 M=1,MNMAXO
 U1(M)=Z(1,1,M)
 V1(M)=Z(2,1,M)
 W1(M)=Z(3,1,M)
 U2(M)=Z(1,2,M)
 V2(M)=Z(2,2,M)
 W2(M)=Z(3,2,M)
 1 IF(IBCINL.LT.0) GO TO 100
 CALL PHIBET(1)
 DO 2 M=1,MNMAX
 BX1(M)=BX3(M)
 BT1(M)=BT3(M)
 BXT1(M)=BXT3(M)
 2 BE1(M)=BE3(M)
 CALL TEAETA(1)
 DO 3 M=1,MNMAX
 EXX1(M)=EXX3(M)

```

ETT1(M)=ETT3(M)
ETX1(M)=ETX3(M)
EXT1(M)=EXT3(M)
EX1(M)=EX3(M)
3 ET1(M)=ET3(M)
3
102 CALL PHIBET(2)
DO 4 M=1,MNMAX
BX2(M)=BX3(M)
BT2(M)=BT3(M)
BXT2(M)=BXT3(M)
4 BE2(M)=BE3(M)
CALL TEAETA(2)
DO 5 M=1,MNMAX
EXX2(M)=EXX3(M)
ETT2(M)=ETT3(M)
ETX2(M)=ETX3(M)
EXT2(M)=EXT3(M)
EX2(M)=EX3(M)
5 ET2(M)=ET3(M)
CALL PHIBET(3)
CALL TEAETA(3)
IF(IBCINL.LT.0) GO TO 20
B1=B(1)
GAM1=GAM(1)
DO 8 M=1,MNMAX
FFS(1,M)=-TT(1,M)*ALOAD+OSE*(BX1(M)+BE1(M)+NU*(BT1(M)+BE1(M)))*B1
FFS(2,M)=OSE*(B1*D1 *BX1(M)+EX1(M)+ET1(M))
FFS(3,M)= LAM2*GAM1 *MT(1,M)*ALOAD-(EXX1(M)+ETX1(M))*SOE
8 FFS(4,M)=C.
DO 9 I=1,4
9 EL1S(I)=ALOAD*EL1(I)
CALL FORCE(1)
20 CALL FORCE(2)
DO 10 K=3,KLL
KP=K+1
CALL UPDATE
CALL PHIBET(KP)
CALL TEAETA(KP)
10 CALL FORCE(K)

```

```

CALL UPDATE
IF(IBCFLN.LT.0) GO TO 120
CALL PHIBET(KMAX)
CALL TEAETA(KMAX)
CALL FORCE(KL)
CALL FORCE(KMAX)
DO 12 I=1,4
  ELLS(I)=ALOAD*ELL(I)
  BL=B(KMAX)
  GAML=GAM(KMAX)
  FLS(4)=0.
  DO 14 M=1,MNMAX
    FLS(1) =-TT(KMAX,M)*ALOAD+OSE*(BX3(M)+BE3(M)+NU*(BT3(M)+BE3(M)
    )+BL
    )*BL
    FLS(2) =OSE*(BL*D1*BXT3(M)+EX3(M)+ET3(M))
    FLS(3) =LAM2*GAML*D1*MT(KMAX,M)*ALOAD-(EXX3(M)+ETX3(M))*SOE
  DO 14 I=1,4
    SUMZ=0.
    DO 15 J=1,4
      SUMZ=SUMZ+ZF1M(I,J,M)*ELLS(J)+ZF2M(I,J,M)*X(J,KMAX,M)
      1 +ZF3M(I,J,M)*X(J,KL,M)+ZF4M(I,J,M)*FLS(J)
    14 Z(I,KMAX2,M)=SUMZ
    LS=1
    150 DO 16 M=1,MNMAX
      DO 16 L=LS,KMAX
        K=KMAX2-L
        KPX=K-1
        KZ=K+1
        DO 17 I=1,4
          SUMZ=0.
          DO 18 J=1,4
            IJ=I+4*(J-1)
            SUMZ=SUMZ-P(IJ,KPX,M)*Z(J,KZ,M)
            ASUMZ=ABS(SUMZ)
            IF(NCONV.NE.1.OR.ASUMZ.LT.1.E-06) GO TO 17
            DELZ=ABS(Z(I,K,M)-SUMZ)
            ZTEST=EPS*ASUMZ
            IF(DELZ.GT.ZTEST) NCONV=0

```

```

17 Z(I,K,M)=SUMZ
16 CONTINUE
  IF(IBCINL.LT.0) GO TO 30
  DO 25 M=1,MNMAX
    CALL HJ(I,M)
    CALL EFG(I,M)
    CALL ABC
    DO 21 I=1,4
      SUMZ=0.
      DO 22 J=1,4
        SUMZ=SUMZ-A(I,J)*Z(J,3,M)-BEE(I,J)*Z(J,2,M)
22    ZT(I)=SUMZ+GEE(I,M)
21    CALL MATINV(C,4,ZT,I,DETERM,IPIVOT,INDEX,4,ISCALE)
    DO 23 I=1,4
23    Z(I,1,M)=ZT(I)
    CONTINUE
25   RETURN
30   CALL INLPOL
DO 101 M=1,MNMAXO
  U1(M)=U2(M)
  V1(M)=V2(M)
  W1(M)=W2(M)
  U2(M)=Z(I,3,M)
  V2(M)=Z(2,3,M)
  W2(M)=Z(3,3,M)
101  GO TO 102
102  CALL FNLPOL
  CALL FORCE(KL)
  IF(M2.EQ.0) GO TO 122
  DO 130 I=1,4
    SUM=0.
    DO 131 J=1,4
      SUM=SUM+CL2(I,J)*X(J,KL,M2)
131  ASUMZ=ABS(SUM)
  IF(NCONV.NE.1.OR.ASUMZ.LT.1.E-06) GO TO 130
  DELZ=ABS(Z(I,KMAX1,M2)-SUM)
  ZTEST=EPS*ASUMZ
  IF(DELZ.GT.ZTEST) NCONV=0
130  Z(I,KMAX1,M2)=SUM

```

```

122 IF(M1.EQ.0) GO TO 123
DO 132 I=1,4
SUM=0.
DO 133 J=1,4
ASUMZ=SUM+CL1(I,J)*X(J,KL,M1)
IF(NCONV.NE.1 .OR. ASUMZ.LT. 1.E-06) GO TO 132
DELZ=ABS(Z(I,KMAX1,M1)-SUM)
ZTEST=EPS*ASUMZ
IF(DELZ.GT.ZTEST) NCONV=0
132 Z(I,KMAX1,M1)=SUM
123 IF(M0.EQ.0) GO TO 124
DO 134 I=1,4
SUM=0.
DO 135 J=1,4
SUM=SUM+CLO(I,J)*X(J,KL,M0)
ASUMZ=ABS(SUM)
IF(NCONV.NE.1 .OR. ASUMZ.LT.1.E-06) GO TO 134
DELZ=ABS(Z(I,KMAX1,M0)-SUM)
ZTEST=EPS*ASUMZ
IF(DELZ.GT.ZTEST) NCONV=0
134 Z(I,KMAX1,M0)=SUM
124 LS=2
GO TO 150
END

```

```

C SUBROUTINE INLPOL--CALCULATES THE BETAS AND ETAS NEEDED AT AN
C INITIAL POLE TO CALCULATE THE DERIVATIVES OF BETAS AND ETAS USED
C IN X AT K=2.
COMMON /BL1/MNMAX/BL3/M0,M1,M2,M3/BL17/DEL/BL6/Z^(4,22,10),SOE/
1BL7/D1,S1/BL11/OMXI(20),PHEE,T0,T2/BL2/B(20)/BL29/BX1(10),BT1(10),
2BX1(10),BE1(10)/BL30/EXX1(10),ETT1(10),ETX1(10),EXT1(10),EX1(10),
3ET1(10)
DO 1 MN=1,MNMAX
BX1(MN)=0.
BT1(MN)=0.
BXT1(MN)=0.
BE1(MN)=0.
EX1(MN)=0.
ET1(MN)=0.
ETX1(MN)=0.
EXX1(MN)=0.
1 IF(M1.EQ.0) RETURN
PHEE=(1.5*Z(3,2,M1)-2.*Z(3,3,M1)+.5*Z(3,4,M1))/DEL
1 +OMXI(1)*Z(1,2,M1)
BET=.5*PHEE**2
T2=0.
IF(M2.EQ.0) GO TO 2
T2=B(1)*D1*((-1.5*Z(1,2,M2)+2.*Z(1,3,M2)-.5*Z(1,4,M2))/DEL
1 +.5*SOE*BET)
Q1=.5*PHEE*T2
BX1(M2)=BET
BT1(M2)=-BET
BXT1(M2)=-BET
ETX1(M1)=Q1
IF(M3.EQ.0) GO TO 2
EXX1(M3)=Q1
ETX1(M3)=-Q1
2 T0=0.
IF(M0.EQ.0) GO TO 3
BX1(M0)=BET
BT1(M0)=BET
T0=B(1)*S1*((-1.5*Z(1,2,M0)+2.*Z(1,3,M0)-.5*Z(1,4,M0))/DEL
1 +OMXI(1)*Z(3,2,M0)+.5*SOE*BET)
3 EXX1(M1)=PHEE*(T0+.5*T2)
RETURN
END

```

SUBROUTINE FNLPOL

```

C SUBROUTINE FNLPOL--CALCULATES THE BETAS AND ETAS NEEDED AT A
C FINAL POLE TO CALCULATE THE DERIVATIVES OF BETAS AND ETAS USED
C IN X AT K=NMAX-1.
COMMON /IBL4/KMAX,KL/IBL12/KMAX1/IBL3/M0,M1,M2,M3/IBL1/MNMAX/BL17
1/DEL/BL6/Z(4,22,10),SOE/BL7/D1,S1/BL11/OMX1(20)*PHEE,T0,T2/BL2/B
2(20)/BL27/BX3(10),BT3(10),BXT3(10),BE3(10)/BL28/EXX3(10),ETT3(10),
3ETX3(10),EXT3(10),EX3(10),ET3(10)

I=KMAX1
J=KMAX
DO 1 MN=1,MNMAX
BX3(MN)=0.
BT3(MN)=0.
BXT3(MN)=0.
BE3(MN)=0.
EX3(MN)=0.
ET3(MN)=0.
ETX3(MN)=0.
1 EXX3(MN)=0.
IF(M1.EQ.0) RETURN
PHEE=-(1.5*Z(3,I,M1)-2.*Z(3,J,M1)+.5*Z(3,KL,M1))/DEL
1 +OMX1(J)*Z(1,I,M1)
BET=.5*PHEE**2
T2=0.
IF(M2.EQ.0) GO TO 2
T2=B(J)*D1*(( 1.5*Z(1,I,M2)-2.*Z(1,J,M2)+.5*Z(1,KL,M2))/DEL
1 +.5*SOE*BET)
Q1=.5*PHEE*T2
BX3(M2)=BET
BT3(M2)=-BET
BXT3(M2)=-BET
ETX3(M1)=Q1
IF(M3.EQ.0) GO TO 2
EXX3(M3)=Q1
ETX3(M3)=-Q1
2 T0=0.
1 IF(M0.EQ.0) GO TO 3
BX3(M0)=BET
BT3(M0)=BET

```

```
T0=B(J)*S1*(( 1.5*Z(1,I,M0)-2.*Z(1,J,M0)+.5*Z(1,KL,M0))/DEL
1   +OMXI(J)*Z(3,I,M0)+.5*SOE*BET)
3 EXX3(M1)=PHEE*(T0+.5*T2)
RETURN
END
```

SUBROUTINE MODES

C SUBROUTINE MODES--CALCULATES THE NEW MODES GENERATED BY THE
 C NONLINEAR QUANTITIES AND SETS UP THE PROPER MODE NUMBER
 C COMBINATIONS FOR CALCULATING THE BETAS AND ETAS.
 COMMON /IBL9/MAXM/IBL1/MNMAX/IBL2/N(10),MNINIT/IBL7/MNMAXO,MAXD(
 110),MAXS(10),MAXY(10),IS(5,10),JS(5,10),ID(10,10),JD(10,10),IJS
 2(10)/IBL11/ICORFL,IPASS
 IF(MAXM.EQ.1) RETURN
 DO 1 MN=1,MNMAXO
 NMN=N(MN)
 NNS=MN
 IF(MNINIT.GT.MN) NNS=MNINIT
 DO 1 MM=NNS,MNMAXO
 NMM=N(MM)
 NTEST=IABS(NMN-NMM)
 DO 2 MMFT=1,MNMAX
 2 IF(NTEST.EQ.N(MMFT)) GO TO 10
 IF(ICORFL.EQ.1) GO TO 1
 MNMAX=MNMAX+1
 N(MNMAX)=NTEST
 MMFT=MNMAX
 IF(MNMAX.EQ.MAXM) ICORFL=1
 10 IF(NMN-NMM) 11,1,12
 11 LOCD=MAXD(MMFT)+1
 MAXD(MMFT)=LOCD
 ID(LOCD,MMFT)=MM
 JD(LOCD,MMFT)=MN
 GO TO 1
 12 LOCD=MAXD(MMFT)+1
 MAXD(MMFT)=LOCD
 ID(LOCD,MMFT)=MN
 JD(LOCD,MMFT)=MM
 1 CONTINUE
 DO 301 MN=1,MNMAXO
 NMN=N(MN)
 NNS=MN
 IF(MNINIT.GT.MN) NNS=MNINIT
 DO 301 MM=NNS,MNMAXO
 NMM=N(MM)

```

NTEST=NMN+NMM
DO 302 MMFT=1,MNMAX
302 IF(NTEST.EQ.N(MMFT)) GO TO 310
    IF(ICORFL.EQ.1) GO TO 301
    MNMAX=MNMAX+1
    N(MNMAX)=NTEST
    MMFT=MNMAX
    IF(MNMAX.EQ.MAXM) ICORFL=1
310 IF(NMN.EQ.NMM) GO TO 360
    LOCs=MAXS(MMFT)+1
    MAXS(MMFT)=LOCs
    IS(LOCs,MMFT)=MN
    JS(LOCs,MMFT)=MM
    GO TO 301
360 IF(NMN.EQ.0) GO TO 301
    MAXSY(MMFT)=1
    IJS(MMFT)=MN
    CONTINUE
301 MNINIT=MNMAX0+1
    IF(ICORFL.GT.0) IPASS=IPASS+1
    IF(IPASS.LT.2 .AND. MNINIT.LE.MNMAX) CALL PMATRIX
    RETURN
END

```

SUBROUTINE OUTPUT(TKN,CHAR,SIGO)
 SUBROUTINE OUTPUT--THIS SUBROUTINE CALCULATES THE ADDITIONAL
 OUTPUT QUANTITIES, WRITES THE OUTPUT, AND SETS UP THE INFORMATION
 FOR PLOTTING

```

REAL NU,MT,NX,MTH,MMXT,MTS,KX,KT,KXT,LAM
COMMON /IBL7/MNMAX/IBL8/IBCINL,IBCFNL/IBL4/KMAX,KL/LBL10/IFKE,
INTMAX/IBL7/MNMAX/IBL8/IBL12/KMAX1/IBL8/LSTEP,ITR/IBL8/N(10)/IBL3/MO,
2M1,M2,M3/BL15/NJ,U1(10),V1(10),W1(10),U2(10),V2(10),W2(10),U3(10),
3,V3(10),W3(10)/BL17/DEL/BL19/TH(6)/BL6/Z(4,22,10),SUE,USE,ALOAD/
4BL12/TDLI/BL7/D1,S1/BL11/OMX1(20),PHIE,T0,T2/BL20/DEL0X(20)/BL2/
5B(20),D(20)/BL5/TT(20,10),WT(20,10)/BL10/U,X,UT,KRA,GA,PHIX(10),
6PHIT(10),PHI(10)/BL27/BX3(10),BT3(10),BXT3(10),BE3(10)/BL31/DEL5,
DIMENSION TX(10),TH(10),WT(10),MXT(10),MX(10)
DIMENSION GS(10)
WRITE(6,101) LSTEP,ALOAD,ITR
LAM=TKN/CHAR
ENL=1
ABZ=SIGO*TKN
ABZ3 =ABZ*TKN*TKN/CHAR
ABZN=CHAR*SOE
IF(LSTEP.EQ.1 .AND. ITR.EQ.1) ENL=0.
DD2=1.-NU**2
D2I=1./DD2
DPI=1./SI
DNI=1./DI
TDLSQI=•5/DELSQ
DO 1 MN=1,MNMAXO
U1(MN)=Z(1,1,MN)
U2(MN)=Z(1,2,MN)
V1(MN)=Z(2,1,MN)
V2(MN)=Z(2,2,MN)
W1(MN)=Z(3,1,MN)
W2(MN)=Z(3,2,MN)
1 DO 21 K=1,KMAX
      K1=K+1
      IF(K.EQ.1.AND.IBCINL.LT.0) GO TO 201

```

```

IF(K.EQ.KMAX.AND.I>CFNL.LT..) GO TO 301
CALL PHIBET(K)
DEX=DEONYX(K)
DOXT=OX-CT
GDO=GA*DOXT
BS=B(K)
DS=D(K)
DD2D=DD2*DS
DO 3 MN=1,MNMAXO
EN=N(MN)
ENR=EN*RR
TT=S=TT(K,MN)
EX=(U3(MN)-U1(MN))*TDLI+UX*W2(MN)+ENL*USE*(BX3(MN)+DX3(MN))
ET=ENR*V2(MN)+GA*U2(MN)+CT*W2(MN)+ENL*USE*(BT3(MN)+E3(MN))
EXT=•2*((V3(MN)-VI(MN))*TDLI-ENR*U2(MN))-GA*V2(MN)
1 ENL*SOE*BXT3(MN)
KT=ENR*PHIT(MN)+GA*PHIX(MN)
KXT=5*(ENR*(-PHIX(MN)-GA*W2(MN)+(W3(MN)-W1(MN))*TDLI)
1 + GDC*V2(MN)+OT*(V3(MN)-V1(MN))*TDLI)
2 TX(MN)=BS*(EX+NJ*ET)-TTS
TTH(MN)=BS*(ET+NJ*EX)-TTS
TXT(MN)=BS*D1*EXT
MX(MN)=Z(4,K1,MN)
MTH(MN)=N*MX(MN)+DD2*KMT(K,MN)
MXT(MN)=DS*D1*KXT
QS(MN)=SIG0*TKN*LA0**2*(GA*MX(MN)+(Z(4,K1+1,MN)-Z(4,K,MN))*TDLI
1 +ENR*MXT(MN)-GA*YTH(MN))
MX(MN)=MX(MN)*ABZ3
MTH(MN)=MTH(MN)*ABZ3
MXT(MN)=MXT(MN)*ABZ3
TX(MN)=TX(MN)*ABZ
TTH(MN)=TT(H(MN)*ABZ
TXT(MN)=TXT(MN)*ABZ
PHIX(MN)=PHIX(MN)*SCE
PHIT(MN)=PHIT(MN)*SCE

```

```

PHI(MN)=PHI(MN)*SCE
U1(MN)=U2(MN)*ABZN
U2(MN)=U3(MN)*ABZN
V1(MN)=V2(MN)*ABZN
V2(MN)=V3(MN)*ABZN
W1(MN)=W2(MN)*ABZN
W2(MN)=W3(MN)*ABZN

3 FK=K
      FIFREQ=IFREQ
      KTST=K/IFREQ
      FKTST=KTST
      FKTTEST=FK/FIFREQ-FKTST
      IF(FKTTEST.NE.0.) GO TO 2
      WRITE(6,102) K
      WRITE(6,103)
      WRITE(6,104) ( N(MN),U1(MN),V1(MN),W1(MN),PHIX(MN),
      PHI(MN),PHI(MN),NN=1,MNMAXO)
      1 WRITE(6,105)
      WRITE(6,114) ( N(MN),RMAX(MN) ,NTH(MN),RXT(MN),TX(MN),
      1 TXT(MN),GS(MN),NN =1,MNMAXO)
      1 IF(NTHMAX.EQ.0) GO TO 2
      WRITE(6,106) K
      WRITE(6,107)
      DO 71 NTH=1,NTHMAX
      UF=0.
      VF=0.
      WF=0.
      PXF=0.
      PTF=0.
      PF=0.
      THET=TH(NTH)
      DO 72 NN=1,MNMAXO
      EN=N(MN)
      FC=EN*THET
      SN=SIN(FC)
      CS=COS(FC)

```

```

UF=UF+U1(MN)*CS
VF=VF+W1(MN)*SN
WF=WF+V1(MN)*CS
PXF=PXF+PHIX(MN)*CS
PTF=PTF+PHIT(MN)*SN
PF=PF+PHI(MN)*SN
72 WRITE(6,108) THET,UF,VF,WF,PXF,PTF,PF
    WRITE(6,109)
DO 73 NTH=1,NTHMAX
AMX=0.
AMTH=0.
AMXT=0.
ANX=0.
ANTH=0.
ANXTH=0.
AQS = 0.
THET=TH(NTH)
DO 74 MN=1,MNMAXO
EN=N(MN)
FC=EN*THET
SN=SIN(FC)
CS=COS(FC)
AMX=AMX+MX(MN)*CS
AMTH=AMTH+TH(MN)*CS
AMXTH=AMXTH+NXT(MN)*SN
ANX=ANX+TX(MN)*CS
ANTH=ANTH+TH(MN)*CS
AQS = AQS + QS(MN)*CS
ANXTH=ANXTH+TXT(MN)*SN
74 ANXTH=ANXTH+TXT(MN)*SN
73 WRITE(6,108) THET,AMX,AMTH,ANXTH,ANTH,ANX,ANXTH,AQS
    GO TO 2
201 WRITE(6,110)
DO 202 MN=1,MNMAXO
    U1(MN) = U2(MN)*ABZN
    V1(MN) = V2(MN)*ABZN
    W1(MN) = W2(MN)*ABZN

```

```

U2(MN) = Z(1,3,MN)*ABZN
V2(MN) = Z(2,3,MN)*ABZN
W2(MN) = Z(3,3,MN)*ABZN
PHIX(MN)=0.
PHIT(MN)=0.
PHI(MN)=0.
MX(MN)=Z(4,2,MN)*ABZ3
MTH(MN)=0.
MXT(MN)=0.
TX(MN)=0.
TTH(MN)=0.
202 TXT(MN)=0.
IF(M1.EQ.0) GO TO 203
CALL INLPOL
PHIX(V1) = PHEE*SCE
PHIT(M1) = -PHEE*SCE
IF(V0.EQ.0) GO TO 204
TX(MC) = T0*ABZ
TTH(W0) = T0*ABZ
MTH(MC) = NX(MC)
204 IF(M2.EQ.0) GO TO 205
TX(V2) = T2*ABZ
TTH(M2) = -T2*ABZ
MTH(V2) = -MX(M2)
MXT(M2) = MTH(M2)
GO TO 205
203 IF(V0.EQ.0) GO TO 206
TX(V0) = S(1)*S1*((-1.2*X1(V0))
+QWX1(1)*W1(MC))*ABZ
1 TTH(W0) = TX(W0)
MTH(MC) = MX(MC)
206 IF(M2.EQ.0) GO TO 205
TX(M2) = E(1)*D1*(-1.2*X1(M2)+2.*Z(1,3,M2)-0.5*X(1,4,M2))/DEL
TX(M2) = TX(M2)*ABZ
TTH(M2) = -TX(M2)
TXT(M2) = -TX(M2)
MTH(M2) = -VX(V2)
MXT(M2) = -VX(M2)
205 WRITE(6,103)

```

```

      WRITE(6,104)(N(MN),I1(MN),V1(MN),V1(MN),PRINT(MN)),
1      PHIT(VN),PHI(KN),MN=1,MNMAX)
      WRITE(6,115)
      WRITE(6,104)(N(MN),V1(MN),V1(MN),TX(MN),TT(MN),TTH(MN)),
1      TXT(MN),MN=1,MNMAX)
1      GO TO 2
      WRITE(6,111)
      DO 302 MN=1,MNMAX
      PHI(X(VN))=C.
      PHIT(MN)=C.
      PHI(VN)=C.
      VVX(VN)=Z(4,KMAX1,MN)*ABZ
      MTH(MN)=C.
      MXT(MN)=C.
      TX(MN)=C.
      TTH(MN)=C.
      302 TXT(VN)=C.
      IF(M1.EQ.C.) GO TO 303
      CALL FNLPOL
      PHI(X(M1)) = PHEE*S0E
      PHIT(M1) = -PHEE*S0E
      IF(M0.EQ.C.) GO TO 304
      TX(MC) = TC*ABZ
      TTH(MC) = TC*ABZ
      MTH(M0) = MX(M0)
      304 IF(M2.EQ.C.) GO TO 305
      TX(M2) = T2*ABZ
      TTH(M2) = -T2*ABZ
      TXT(M2) = -T2*ABZ
      MTH(V2) = -MX(M2)
      MXT(M2) = MTH(V2)
      GO TO 305
303 IF(MC.EQ.C.) GO TO 306
      TX(M0) = B(KMAX)*DP1*((1.5*U2(MC)-2.*Z(L,KMAX,M0)+.5*Z(L,KL,M0))
1      /DEL+OMX1(KMAX)*W2(MC))*ABZ
      TTH(M0) = TX(M0)
      MTH(MC) = MX(M0)
306 IF(M2.EQ.C.) GO TO 305

```

```

TX(M2)=B(KMAX)*D1*(1.5*U2(M2)-2.*Z(1,KMAX,M2)+.5*Z(1,KL,M2))/DEL
TX(M2) = TX(M2)*ABZ
TTT(M2) = -TX(M2)
TXT(M2) = -TX(M2)
MTH(M2) = -MX(N2)
MXT(M2) = -MX(N2)
305 WRITE(6,103)
      WRITE(6,104)(N(MN),U2(MN),V2(NN),W2(MN),PHIX(NN),PHIT(MN),
      PHI(MN),MN=1,MNMAXC)
1      WRITE(6,115)
      WRITE(6,104)(N(MN),MX(NN),MTH(NN),VXT(NN),TX(NN),TTT(NN),
      TXT(MN),KN=1,KNMAXC)
1      GO TO 2
101 FORMAT(1H1,7H LSTEP=I2/7H ALLOAD=E16.7/5H ITR=I2// //)
102 FORMAT(1H1,8HSTATION I3,14H-NUDAL OUTPUT/ )
103 FORMAT(
1V   14H   W
      14H   PHI
      2   PHI   /
104 FORMAT(15,9X,1P6E14.4)
105 FORMAT(1H1,
      1  MTH   14H   MSTH   14H   N
      2H   TSTH   14H   QS   14H   TS
106 FORMAT(1H1,8HSTATION I3,14H- THETA OUTPUT/ )
107 FORMAT(
      1V   14H   THETA   14H   U
      2   PHI   /
108 FORMAT(1P8E14.4)
109 FORMAT(1H1,
      1  MTH   14H   MSTH   14H   THETA
      2H   NSTH   14H   QS   14H   NS
110 FORMAT(13H INITIAL POLE/ )
111 FORMAT(11H FINAL POLE/ )
114 FORMAT(15,9X,1P7E14.4)
115 FORMAT(1H1,
      1  MTH   14H   MSTH   14H   N
      2H   TSTH   /)   14H   NS
14H   TTH   14H   NS

```

```
2 CONTINUE
DO 979 MN=1,MNMAXO
U1(MN)=U1(MN)/ABZN
U2(MN)=U2(MN)/ABZN
V1(MN)=V1(MN)/ABZN
V2(MN)=V2(MN)/ABZN
W1(MN)=W1(MN)/ABZN
W2(MN)=W2(MN)/ABZN
979
21 CONTINUE
RETURN
END
```

```

C
SUBROUTINE PHIBET(K)
SUBROUTINE PHIBET--CALCULATES THE PHIS AND THE BETAS.
COMMON /IBL1/MNMAXX/IBL7/MNMAXO,MAXD(10),MAXS(10),MAXSY(10),IS(5,
110),JS(5,10),ID(10,10),JD(10,10),IJS(10)/IBL2/N(10)/BL6/Z(4,22,10
2)/BL12/TDL1/BL8/R(20),GAM(20),OMT(20)/BL11/OMXI(20)/BL15/NU,U1(10
3),V1(10),W1(10),U2(10),V2(10),W2(10),U3(10),V3(10),W3(10)/BL10/OX
4,OT,RRA,GA,PHIX(10),PHIT(10),PHI(10)/BL27/BX3(10),BT3(10),BXT3(10
5),BE3(10)

OX=OMXI(K)
OT=OMT(K)
RRA=1.0/R(K)
GA=GAM(K)
KP2=K+2
DO 1 M=1,MNMAXO
EN=N(M)
U3(M)=Z(1,KP2,M)
V3(M)=Z(2,KP2,M)
W3(M)=Z(3,KP2,M)
PHIX(M)=-TDL1*(W3(M)-W1(M))+OX*U2(M)
PHIT(M)=EN*W2(M)*RRA+V2(M)*OT
1 PHI(M)=(TDL1*(V3(M)-V1(M))+GA*V2(M)+EN*U2(M)*RRA)*.5
DO 9 M=1,MNMAX
SMO=0.
SMT=0.
SMR=0.
SMF=0.
IF(N(M).EQ.0) GO TO 20
MAXL=MAXS(M)
IF(MAXL.EQ.0) GO TO 2
DO 3 L=1,MAXL
I=IS(L,M)
J=JS(L,M)
SMO=SMO+PHIX(I)*PHIX(J)
SMT=SMT-PHIT(I)*PHIT(J)
SMR=SMR+PHIX(I)*PHIT(J)+PHIX(J)*PHIT(I)
3 SMF=SMF-PHI(I)*PHI(J)
2 MAXL=MAXD(M)
IF(MAXL.EQ.0) GO TO 4
DO 5 L=1,MAXL

```

```

I=ID(L,M)
J=JD(L,M)
SMO=SMO+PHIX(I)*PHIX(J)
SMT=SMT+PHIT(I)*PHIT(J)
SMR=SMR-PHIX(I)*PHIT(J)+PHIX(J)*PHIT(I)
5 SMF=SMF+PHI(I)*PHI(J)
4 IF(MAXSY(M).EQ.0) GO TO 10
I=JS(M)
SMO=SMO+PHIX(I)**2/2.
SMT=SMT-PHIT(I)**2/2.
SMR=(SMR+PHIX(I)*PHIT(I))
SMF=SMF-PHI(I)**2/2.
GO TO 10
20 DO 21 L=1,MNMAXO
SMO=SMO+PHIX(L)**2
SMT=SMT+PHIT(L)**2
21 SMF=SMF+PHI(L)**2
IF(M.GT.MNMAXO) GO TO 11
SMO=SMO+PHIX(M)**2
11 BX3(M)=SMO*.5
BT3(M)=SMT*.5
BE3(M)=SMF*.5
BXT3(M)=0.
GO TO 9
10 BX3(M)=SMO
BT3(M)=SMT
BXT3(M)=SMR*.5
BE3(M)=SMF
9 CONTINUE
RETURN
END

```

C
 SUBROUTINE TEAETA(K)
 SUBROUTINE TEAETA--CALCULATES THE INPLANE FORCES AND THE ETAS.
 REAL NU

```

COMMON /IBL1/MNMAX/IBL7/MNMAXO,MAXD(10),MAXSY(10),IS(5,
110),JS(5,10),ID(10,10),JD(10,10),IJS(10)/IBL2/N(10)/BL15/NU,U1(10
2),V1(10),W1(10),U2(10),V2(10),W2(10),U3(10),V3(10),W3(10)/BL6/Z(
34,22,10),SOE,OSE,BL12/TDL1/BL7/D1/BL2/B(20),D(20)/BL5/TT(20,10)/
4BL10/OX,OT,RRA,GA,PHIX(10),PHIT(10),PHI(10)/BL27/BX3(10),BT3(10),
5BXT3(10),BE3(10)/BL28/EXX3(10),ETT3(10),ETX3(10),EXT3(10),EX3(10)
6,ET3(10)

DIMENSION TX(10),TTH(10),TXT(10)

BS=B(K)
DS=D(K)
DO 1 M=1,MNMAXO
EN=N(M)
TTS=TT(K,M)
EX=(U3(M)-U1(M))*TDL1+OX*W2(M)+OSE*(BX3(M)+BE3(M))
ET= EN *V2(M)*RRA+GA*U2(M)+OT*W2(M)+OSE*(BT3(M)+BE3(M))
EXT=.5*(TDL1*(V3(M)-V1(M))- EN *U2(M)*RRA-GA*V2(M))+OSE*BXT3(M)
TX(M)=BS*(EX+NU*ET)-TTS
TTH(M)=BS*(ET+NU*EX)-TTS
1 TXT(M)=BS*D1*EXT
DO 9 M=1,MNMAX
SMF=0.
SMS=0.
SME=0.
SMN=0.
SMT=0.
IF(N(M).EQ.0) GO TO 20
MAXL=MAXS(M)
IF(MAXL.EQ.0) GO TO 2
DO 3 L=1,MAXL
I=IS(L,M)
J=JS(L,M)
SMF=SMF+TX(I)*PHIX(J)+TX(J)*PHIX(I)
SMS=SMS+TTH(I)*PHIT(J)+TTH(J)*PHIT(I)
SMV=SMV-PHIT(I)*TXT(J)-PHIT(J)*TXT(I)

```

```

SME=SME+PHIX(I)*TXT(J)+PHIX(J)*TXT(I)
SMN=SMN+TX(I)*PHI(J)+TX(J)*PHI(I)
3 SMT=SMT+TH(I)*PHI(J)+TH(J)*PHI(I)
2 MAXL=MAXD(M)
1 IF(MAXL.EQ.0) GO TO 4
DO 5 L=1,MAXL
I=ID(L,M)
J=JD(L,M)
SMF=SMF+TX(I)*PHIX(J)+TX(J)*PHIX(I)
SMS=SMS-TH(I)*PHIT(J)+TH(J)*PHIT(I)
SMV=SMV+PHIT(I)*TXT(J)+PHIT(J)*TXT(I)
SME=SME-PHIX(I)*TXT(J)+PHIX(J)*TXT(I)
SMN=SMN-TX(I)*PHI(J)+TX(J)*PHI(I)
5 SMT=SMT-TH(I)*PHI(J)+TH(J)*PHI(I)
4 IF(MAXSY(M).EQ.0) GO TO 10
I=IJS(M)
SMF=SMF+TX(I)*PHIX(I)
SMS=SMS+TH(I)*PHIT(I)
SMV=SMV-PHIT(I)*TXT(I)
SME=SME+PHIX(I)*TXT(I)
SMN=SMN+TX(I)*PHI(I)
SMT=SMT+TH(I)*PHI(I)
GO TO 10

```

```

20 DO 21 L=1,MNMAXO
   SMF=SMF+TX(L)*PHIX(L)
21 SMV=SMV+PHIT(L)*TXT(L)
   IF(M.GT.MNMAXO) GO TO 10
   SMF=SMF+TX(M)*PHIX(M)
10 EXX3(M)=SMF**.5
   ET3(M)=SMS**.5
   ETX3(M)=SMV**.5
   EXT3(M)=SME**.5
   EX3(M)=SMN**.5
   ET3(M)=SMT**.5
9 CONTINUE
   DO 30 M=1,MNMAXO
      U1(M)=U2(M)
      V1(M)=V2(M)
      W1(M)=W2(M)
      U2(M)=U3(M)
      V2(M)=V3(M)
      W2(M)=W3(M)
30 RETURN
END

```

```

C SUBROUTINE FORCE--THIS SUBROUTINE CALCULATES THE GEE VECTOR AND
C THE X VECTOR
REAL NU,MT,LAM2
COMMON /IBL1/MNMAX/IBL4/KMAX/IBL8/LSTEP,IIR/IBL2/N(10)/BL15/NU/BL
117/DEL/BL14/LAM2/BL6/Z(4,22,10),SOE,OSE,ALOAD/BL12/TDL1,TDEL/BL7/
2D1/BL8/R(20),GAM(20),OMT(20)/BL11/OMX1(20)/BL2/B(20),D(20),DB(20)
3/BL3/PR(20,10),PX(20,10),PT(20,10)/BL5/TT(20,10),MT(20,10),DTT(20
4,10),DMT(20,10)/BL24/DL(4,4,10),DG(4,4,10),DF(4,4,10)/BL4/P(16,20
5,10),X(4,20,10)/IBL9/MAXM /BL9/FFS(4,10),EL
61S(4),GEES(4,10)/BL27/BX3(10),BT3(10),BXT3(10),BE3(10)/BL28/EXX3(
710),ETT3(10),ETX3(10),EXT3(10),EX3(10),ET3(10)/BL29/BX1(10),BT1(1
80),BXT1(10),BE1(10),BX2(10),BT2(10),BXT2(10),BE2(10)
COMMON /BL30/EXX1(10),ETT1(10),EXT1(10),EX1(10),ET1(10),
1EXX2(10),ETT2(10),EXT2(10),EX2(10),ET2(10)
COMMON DEE(16,20,10),DST(16,20,10)
DIMENSION GEE(4),CEE1(20,10),CEE2(20,10),CEE3(20,10),CEE4(1
120,10)

EQUIVALENCE (CEE1,PX),(CEE2,PT),(CEE3,PR),(CEE4,DTT)
FDIFF(A,B,C)=(-1.5*A+2.*B-.5*C)/DEL
RS=R(K)
RR=1./RS
GA=GAM(K)
OX=OMX1(K)
OT=OMT(K)
DL2=D1*LAM2
BS=B(K)
DBS=DB(K)
K1=K-1
IF(IIR.GT.1.OR. LSTEP.GT.1) GO TO 2
DO 1 M=1,MNMAX
EN=N(M)
ENR= EN *RR
TEE=TT(K,M)
EMT=MT(K,M)
CEE1(K,M)=(-PX(K,M)+DTT(K,M)-DL2*GA*OX*EMT)*TDEL
CEE2(K,M)=(-PT(K,M)-ENR*TEE-DL2*ENR*OT*EMT)*TDEL
CEE3(K,M)=(-PR(K,M)-(OX+OT)*TEE-DL2*(GA*DMT(K,M)-(OX*OT-ENR**2)
*EMT))*TDEL

```

```

1 CEEE4(K,M)=EMT*TDEL
  MN1=MNMAX+1
  DO 3 M=MN1,MNMAX
    CEEE1(K,M)=0.
    CEEE2(K,M)=0.
    CEEE3(K,M)=0.
    3 CEEE4(K,M)=0.
    2 DO 4 M=1,MNMAX
      EN=N(M)
      ENR=EN*RR
      GEE(1)=CEEE1(K,M)*ALOAD
      GEE(2)=CEEE2(K,M)*ALOAD
      GEE(3)=CEEE3(K,M)*ALOAD
      GEE(4)=CEEE4(K,M)*ALOAD
      IF(K.GT.1) GO TO 6
      BX2T=BX1(M)
      BT2T=BT1(M)
      BX2T=BX1(M)
      BE2T=BE1(M)
      EXX2T=EXX1(M)
      ETT2T=ETT1(M)
      ETX2T=ETX1(M)
      EXT2T=EXT1(M)
      EX2T=EX1(M)
      ET2T=ET1(M)
      DBX= FDIFF(BX2T,BX2(M),BX3(M))
      DBT= FDIFF(BT2T,BT2(M),BT3(M))
      DBXT= FDIFF(BXT2T,BXT2(M),BXT3(M))
      DBE= FDIFF(BE2T,BE2(M),BE3(M))
      DET= FDIFF(ET2T,ET2(M),ET3(M))
      DEX= FDIFF(EX2T,EX2(M),EX3(M))
      DEXX= FDIFF(EXX2T,EXX2(M),EXX3(M))
      DETX= FDIFF(ETX2T,ETX2(M),ETX3(M))
      GO TO 7
6 IF(K.LT.KMAX) GO TO 8
  BX2T=BX3(M)
  BT2T=BT3(M)
  BX2T=BXT3(M)
  BE2T=BE3(M)

```

```

EXX2T=EXX3(M)
ETT2T=ETT3(M)
ETX2T=ETX3(M)
EXT2T=EXT3(M)
EX2T=EX3(M)
ET2T=ET3(M)
DBX=-FDIFF(BX2T,BX2(M),BX1(M))
DBT=-FDIFF(BT2T,BT2(M),BT1(M))
DBXT=-FDIFF(BXT2T,BXT2(M),BXT1(M))
DBE=-FDIFF(BE2T,BE2(M),BE1(M))
DET=-FDIFF(ET2T,ET2(M),ET1(M))
DEX=-FDIFF(EX2T,EX2(M),EX1(M))
DEXX=-FDIFF(EXX2T,EXX2(M),EXX1(M))
DETX=-FDIFF(ETX2T,ETX2(M),ETX1(M))
GO TO 7
     8 DBX=TDLI*(BX3(M)-BX1(M))
     DBT=TDLI*(BT3(M)-BT1(M))
     DBE=TDLI*(BE3(M)-BE1(M))
     DBXT=TDLI*(BXT3(M)-BXT1(M))
     DET=TDLI*(ET3(M)-ET1(M))
     DEX=TDLI*(EX3(M)-EX1(M))
     DEXX=TDLI*(EXX3(M)-EXX1(M))
     DETX=TDLI*(ETX3(M)-ETX1(M))
     BX2T=BX2(M)
     BT2T=BT2(M)
     BX2T=BXT2(M)
     BE2T=BE2(M)
     EXX2T=EXX2(M)
     ETT2T=ETT2(M)
     ETX2T=ETX2(M)
     EXT2T=EXT2(M)
     EX2T=EX2(M)
     ET2T=ET2(M)
    7 GEE(1)=GEE(1)-OSE*(BS*(DBX+DBE+GA*D1*(BX2T-BT2T)+NU*(DBT+DBE)
     +ENR*D1*BXT2T)+DBS*(BX2T+BE2T+NU*(BT2T+BE2T))-2.*OX*
     1           (EXX2T+ETX2T)-ENR*(EX2T+ET2T)*TDEL
     2 GEE(2)=GEE(2)+OSE*(BS*(ENR*(BT2T+BE2T+NU*(BX2T+BE2T))-D1*
     1           (DBXT+2.*GA*BXT2T))-D1*DBS*BX2T+2.*OT*(ETT2T+EXT2T)
     2           -(DEX+DET))*TDEL

```

```

GEE(3)=GEE(3)+OSE*(BS*((OX+NU*OT)*(BX2T+BE2T)+(OT+NU*OX)*
1      (BT2T+BE2T))+2*(GA*(EXX2T+ETX2T)+DEXX+DET+ENR*
2      (EXT2T+ETT2T)))*TDEL
IF(K.GT.1) GO TO 10
DO 20 I=1,4
GEE((I,M)=GEE(I)
SUMX=0.
DO 21 J=1,4
21 SUMX=SUMX+DL(I,J,M)*EL1S(J)+DG(I,J,M)*GEE(J)+DF(I,J,M)*FFS(J,M)
20 X(I,1,M)=SUMX
GO TO 4
10 DO 11 I=1,4
11 SUMX=0.
DO 12 J=1,4
12 IJ=I+4*(J-1)
12 SUMX=SUMX+DEE(IJ,K,M)*GEE(J)-DST(IJ,K,M)*X(J,K1,M)
11 X(I,K,M)=SUMX
4 CONTINUE
RETURN
END

```

C SUBROUTINE UPDATE--THIS SUBROUTINE UPDATES THE STORAGE LOCATIONS
 OF THE BETAS AND ETAS
 COMMON /IBL1/MNMAX/BL27/BX3(10),BT3(10),BX3(10),BE3(10)/BL28/EXX
 13(10),ETT3(10),ETX3(10),EXT3(10),EX3(10),ET3(10),BX1(10),BT1
 2(10),BXT1(10),BE1(10),BX2(10),BT2(10),BXT2(10),BE2(10),BL30/EXX1(10),
 ETT1(10),ETX1(10),EXT1(10),EX1(10),ET1(10),EXX2(10),ETT2(10),
 ETX2(10),EXT2(10),EX2(10),ET2(10)
 DO 1 M=1,MNMAX
 BX1(M)=BX2(M)
 BT1(M)=BT2(M)
 BXT1(M)=BXT2(M)
 BE1(M)=BE2(M)
 BX2(M)=BX3(M)
 BT2(M)=BT3(M)
 BXT2(M)=BXT3(M)
 BE2(M)=BE3(M)
 EXX1(M)=EXX2(M)
 ETT1(M)=ETT2(M)
 ETX1(M)=ETX2(M)
 EXT1(M)=EXT2(M)
 EX1(M)=EX2(M)
 ET1(M)=ET2(M)
 EXX2(M)=EXX3(M)
 ETT2(M)=ETT3(M)
 ETX2(M)=ETX3(M)
 EXT2(M)=EXT3(M)
 EX2(M)=EX3(M)
 1 ET2(M)=ET3(M)
 RETURN
 END

SUBROUTINE MATINV(A,N,B,M,DETERM,IPIVOT,INDEX,NMAX,ISCALE)
MATRIX INVERSION WITH ACCOMPANYING SOLUTION OF LINEAR EQUATIONS

C C
C DIMENSION IPIVOT(N),A(NMAX,N),B(NMAX,M),INDEX(NMAX,2)
C EQUIVALENCE (IROW,JROW), (ICOLUMN,JCOLUMN), (AMAX, T, SWAP)
C
C INITIALIZATION
C
C 5 ISCALE=0
C 6 R1=10.0**18
C 7 R2=1.0/R1
C 10 DETERM=1.0
C 15 DO 20 J=1,N
C 20 IPIVOT(J)=0
C 30 DO 550 I=1,N
C
C SEARCH FOR PIVOT ELEMENT
C
C 40 AMAX=0.0
C 45 DO 105 J=1,N
C 50 IF (IPIVOT(J)-1) 60, 105, 60
C 60 DO 100 K=1,N
C 70 IF (IPIVOT(K)-1) 80, 100, 740
C 80 IF (ABS(AMAX)-ABS(A(J,K))) 85, 100, 100
C 85 IROW=J
C 90 ICOLUMN=K
C 95 AMAX=A(J,K)
C 100 CONTINUE
C 105 CONTINUE
C 110 IPIVOT(ICOLUMN)=IPIVOT(ICOLUMN)+1
C
C INTERCHANGE ROWS TO PUT PIVOT ELEMENT ON DIAGONAL
C
C 130 IF (IROW-ICOLUMN) 140, 260, 140
C 140 DETERM=-DETERM
C 150 DO 200 L=1,N
C 160 SWAP=A(IROW,L)

```

170 A(IROW,L)=A(ICOLUMN,L) F4020037
200 A(ICOLUMN,L)=SWAP F4020038
205 IF(M) 260, 260, 210 F4020039
210 DO 250 L=1, M F4020040
220 SWAP=B(IROW,L) F4020041
230 B(IROW,L)=B(ICOLUMN,L) F4020042
250 B(ICOLUMN,L)=SWAP F4020043
260 INDEX(I,1)=IROW F4020044
270 INDEX(I,2)=ICOLUMN F4020045
310 PIVOT=A(ICOLUMN,ICOLUMN)

C SCALE THE DETERMINANT

C
1000 PIVOTI=PIVOT
1005 IF(ABS(DETERM)-R1)1030,1010,1010
1010 DETERM=DETERM/R1
1015 ISCALE=ISCALE+1
1020 DETERM=DETERM/R1
1025 ISCALE=ISCALE+1
GO TO 1060
1030 IF(ABS(DETERM)-R2)1040,1040,1060
1040 DETERM=DETERM*R1
1045 ISCALE=ISCALE-1
1050 DETERM=DETERM*R1
1055 ISCALE=ISCALE-1
1060 IF(ABS(PIVOTI)-R1)1090,1070,1070
1070 PIVOTI=PIVOTI/R1
1075 ISCALE=ISCALE+1
1080 PIVOTI=PIVOTI/R1
1085 ISCALE=ISCALE+1
GO TO 320
1090 IF(ABS(PIVOTI)-R2)2000,2000,320
2000 PIVOTI=PIVOTI*R1
2005 ISCALE=ISCALE-1
2010 PIVOTI=PIVOTI*R1
2015 ISCALE=ISCALE-1

```

320 DETERM=DETERM*PIVOTI

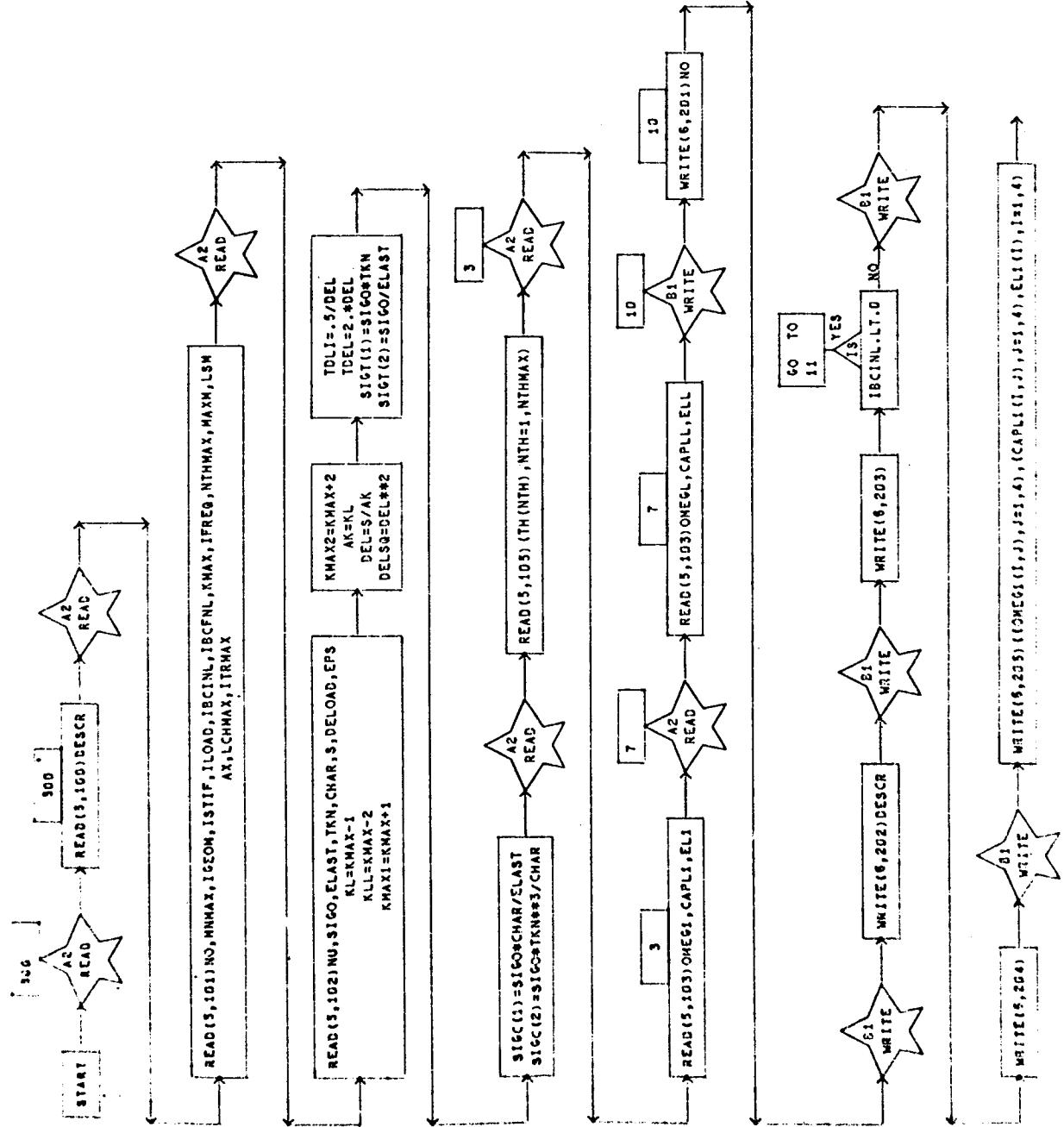
C DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
C 330 A(ICOLUMN,ICOLUMN)=1.0
340 DO 350 L=1,N
350 A(ICOLUMN,L)=A(ICOLUMN,L)/PIVOT
355 IF(M) 380, 360
360 DO 370 L=1,M
370 B(ICOLUMN,L)=B(ICOLUMN,L)/PIVOT
C
C REDUCE NON-PIVOT ROWS
C
C 380 DO 550 L1=1,N
390 IF(L1-1,ICOLUMN) 400, 550, 400
400 T=A(L1,ICOLUMN)
420 A(L1,ICOLUMN)=0.0
430 DO 450 L=1,N
450 A(L1,L)=A(L1,L)-A(ICOLUMN,L)*T
455 IF(M) 550, 550, 460
460 DO 500 L=1,M
500 B(L1,L)=B(L1,L)-B(ICOLUMN,L)*T
550 CONTINUE
C
C INTERCHANGE COLUMNS
C
C 600 DO 710 I=1,N
610 L=N+1-I
620 IF (INDEX(L+1)-INDEX(L,2)) 630, 710, 630
630 JROW=INDEX(L,1)
640 JCOLUMN=INDEX(L,2)
650 DO 705 K=1,N
660 SWAP=A(K,JROW)
670 A(K,JROW)=A(K,JCOLUMN)
700 A(K,JCOLUMN)=SWAP
705 CONTINUE
710 CONTINUE
740 RETURN
END

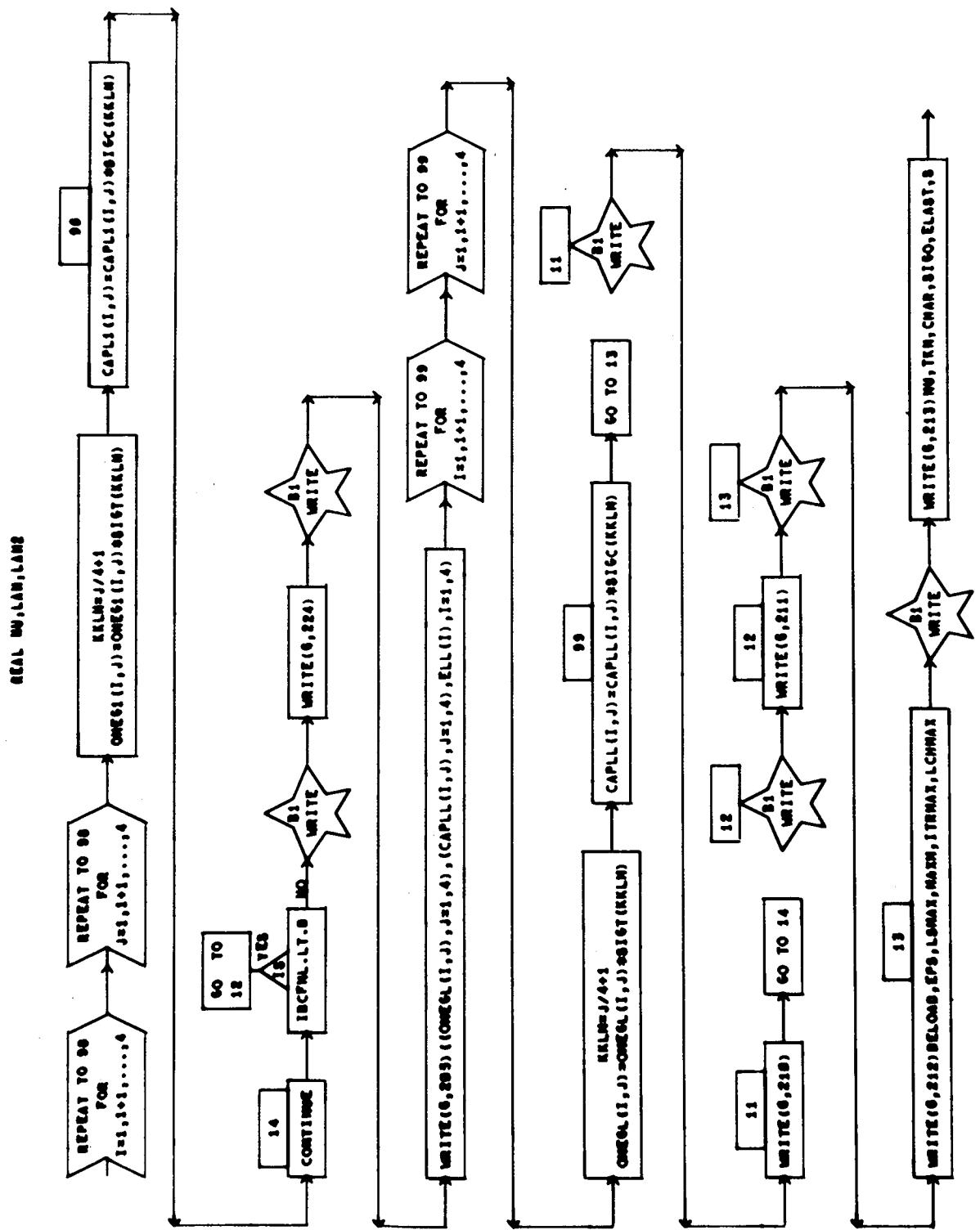
APPENDIX G

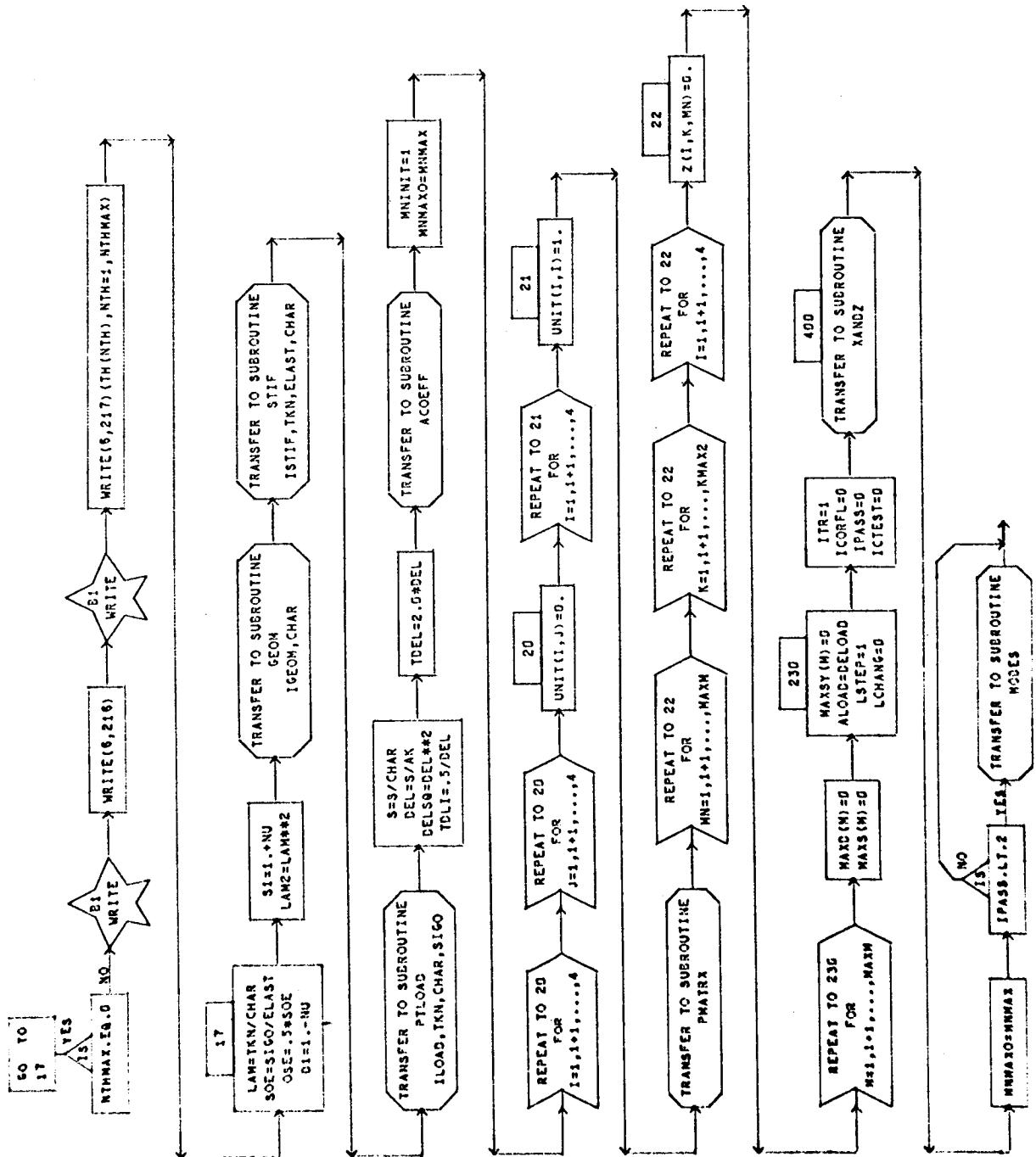
FLOW CHARTS

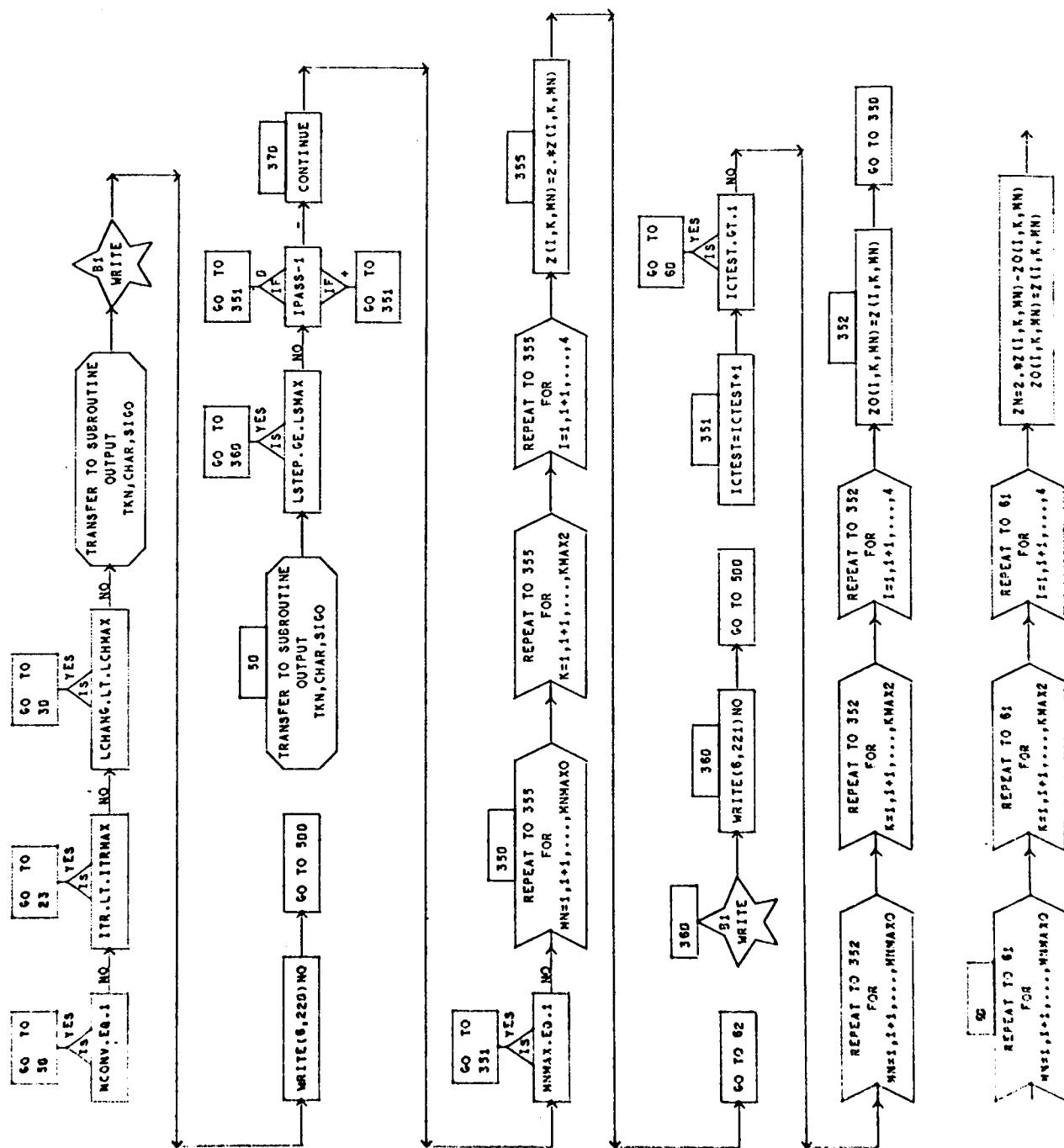
PRECEDING PAGE BLANK NOT FILMED.

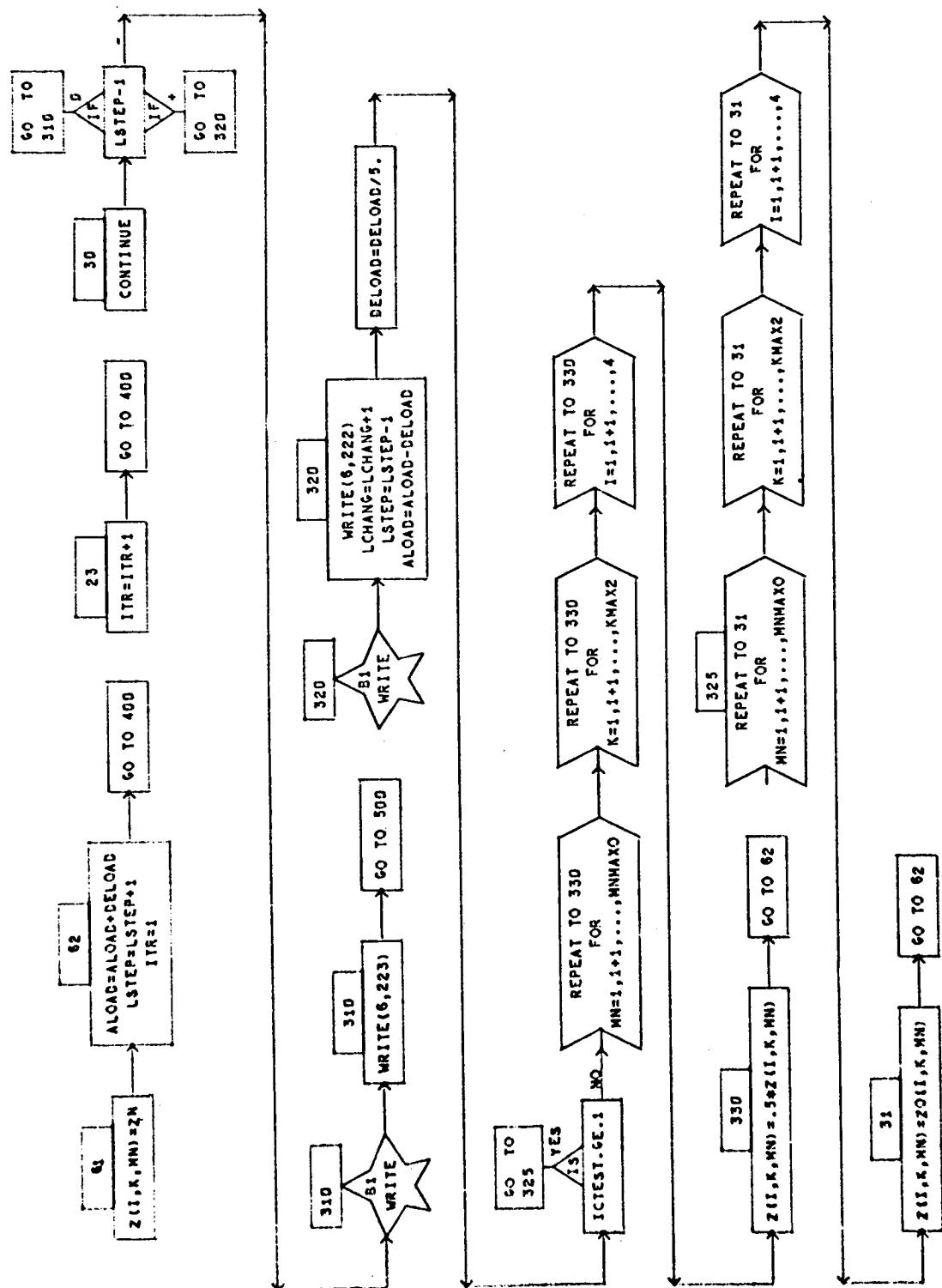
		DIMENSIONED VARIABLES			STORAGES		
SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	
CONST	20	DESCR.	12	SIGT	2	SIGC	2



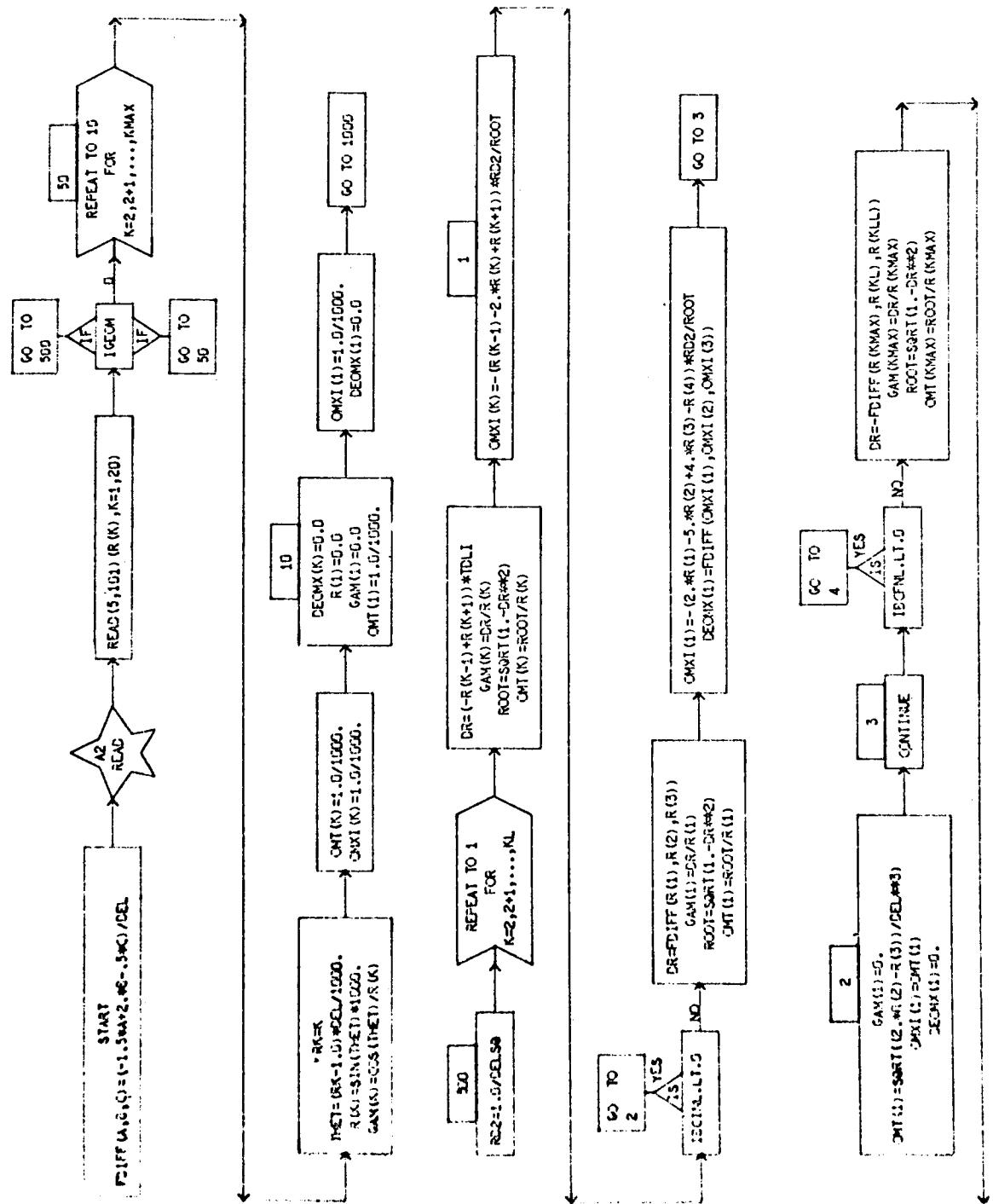


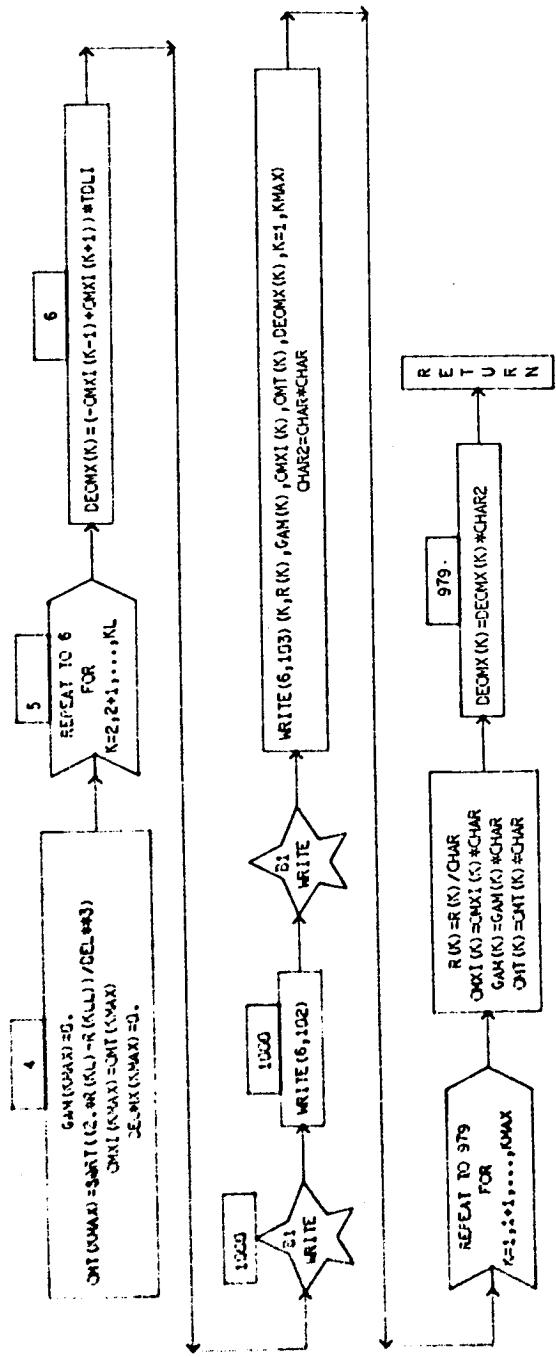






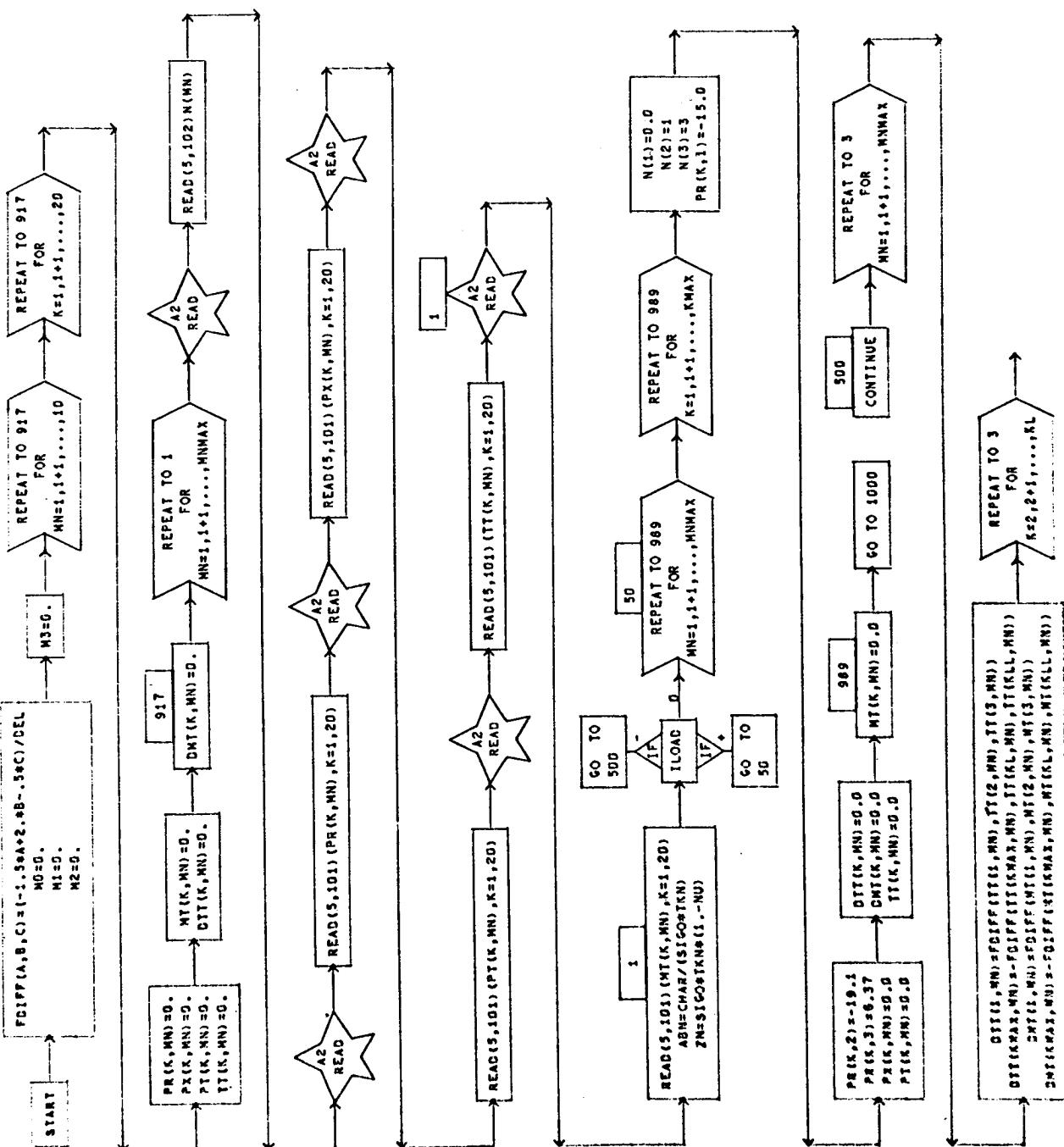
SUBROUTINE GEOM(IGEM, CHAR)





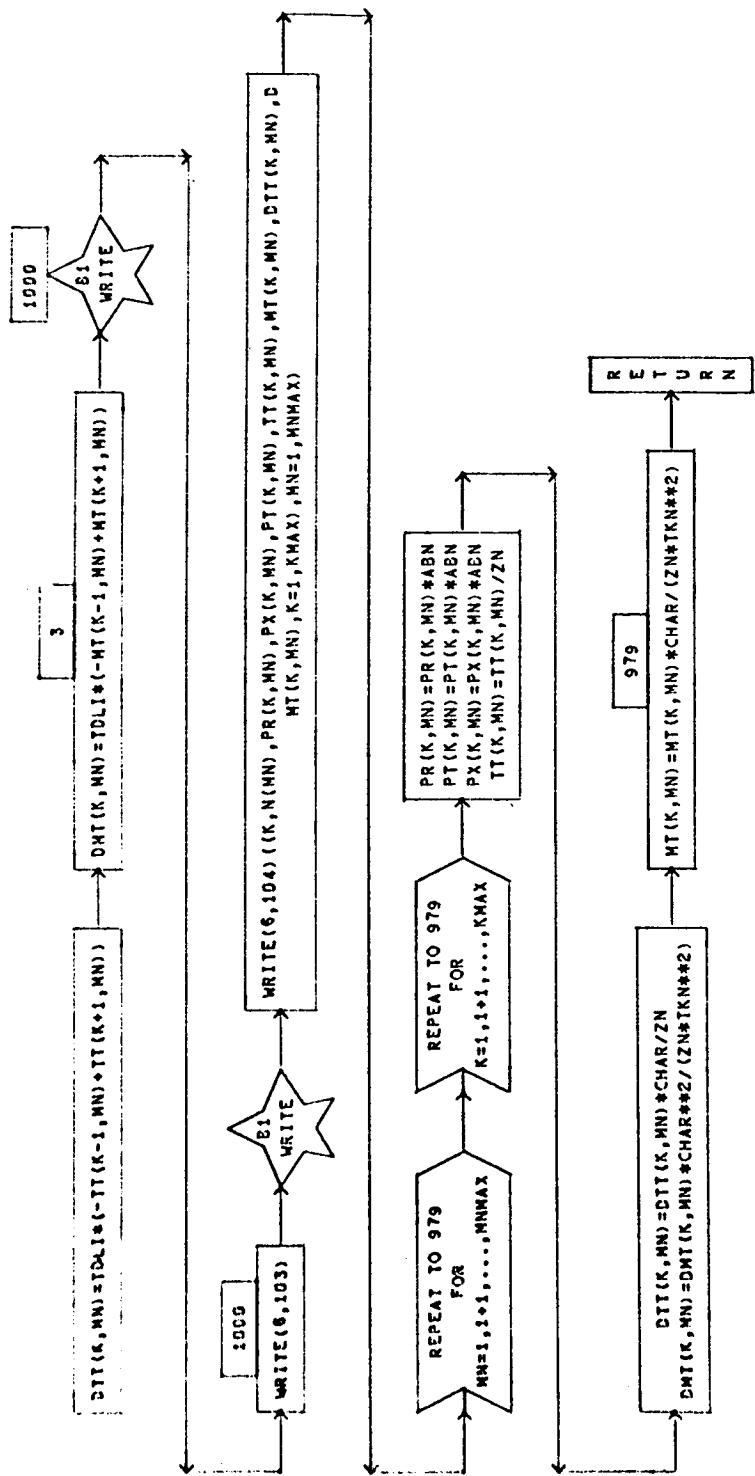
SUBROUTINE PYLOAD(1LOAD,TKN,CHAR,\$160)

PAGE 1

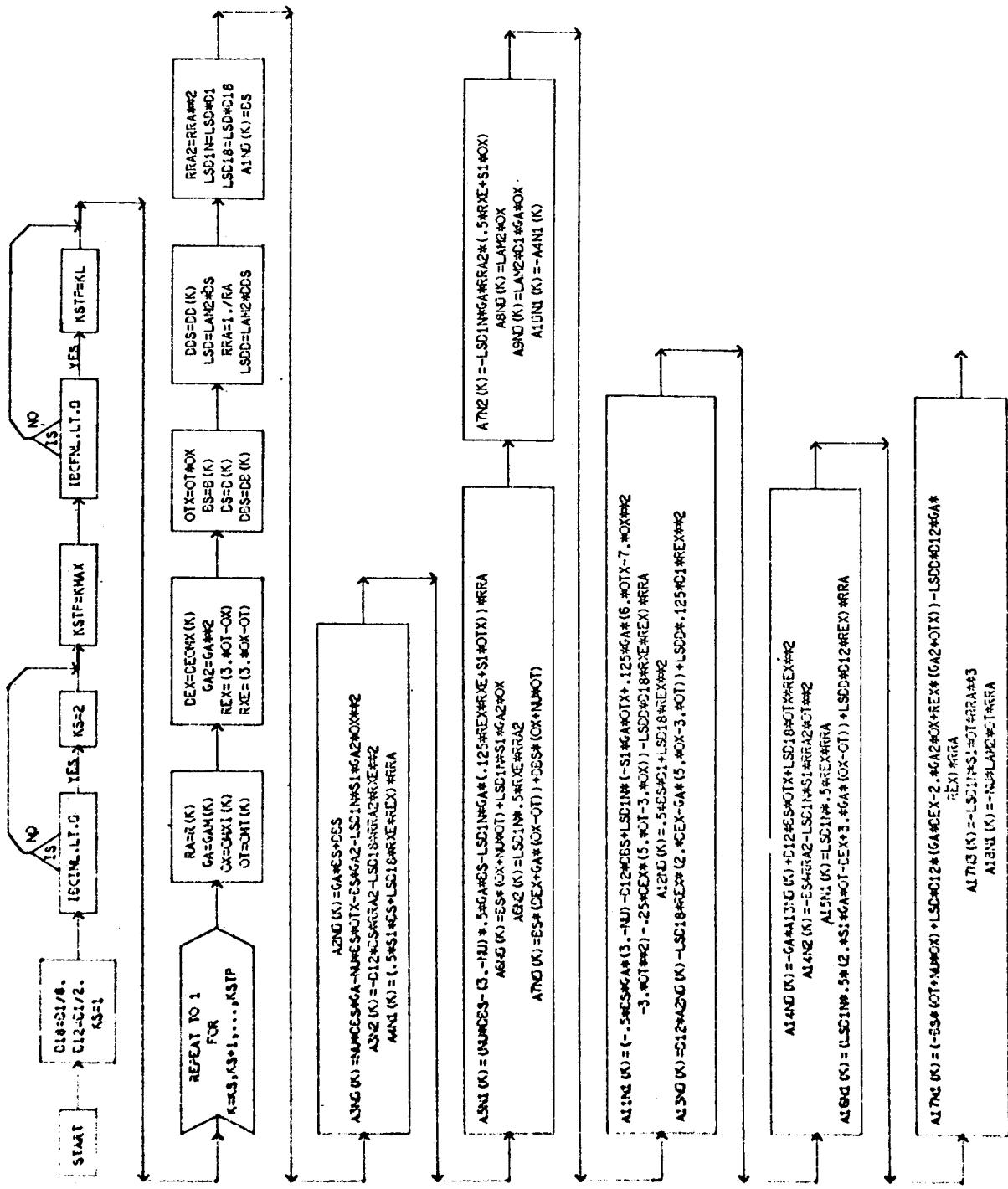


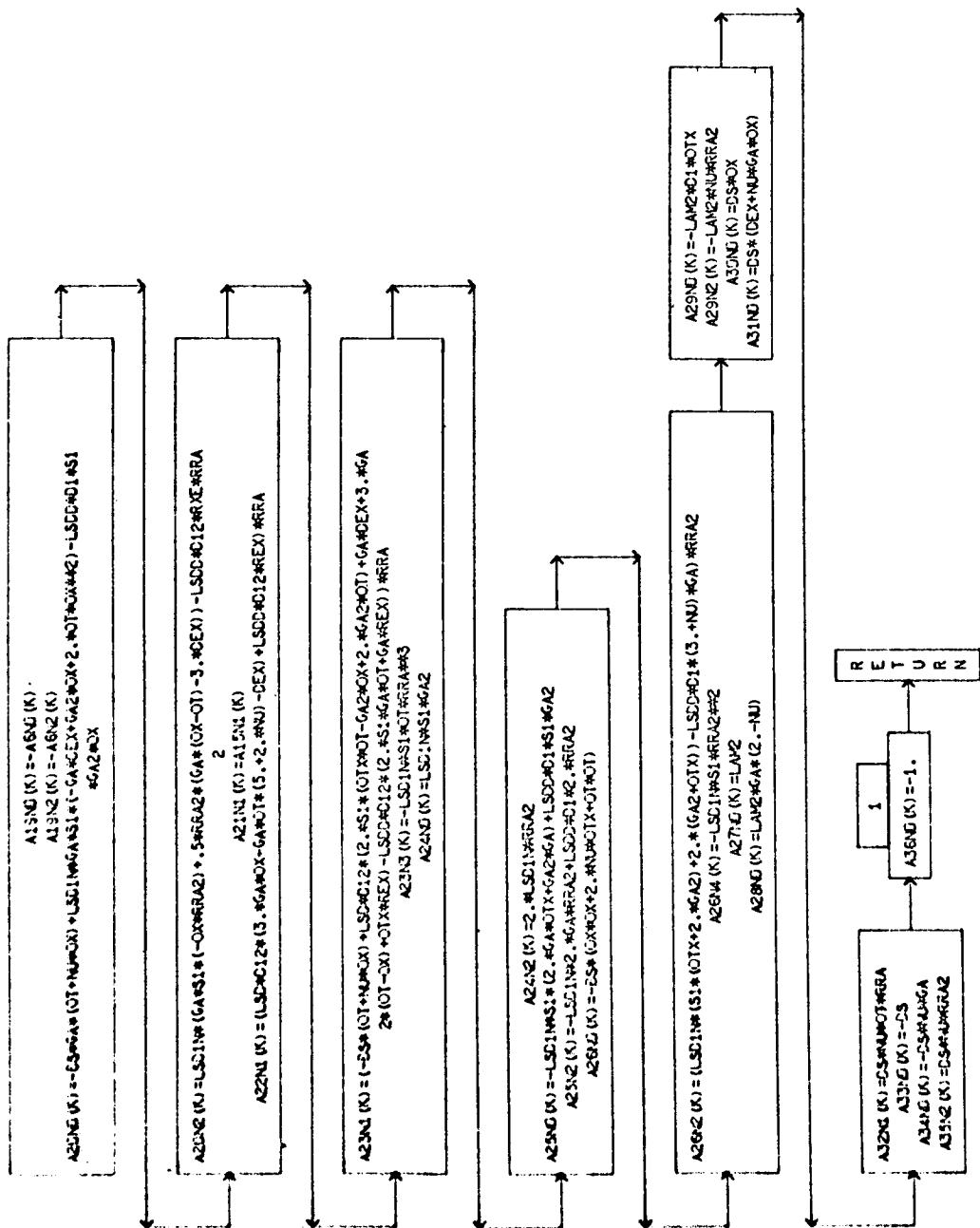
SUBROUTINE PTLOAD(ILLOAD,TKN,CHAR,SIGO)

PAGE 2



SUBROUTINE ACCEFF

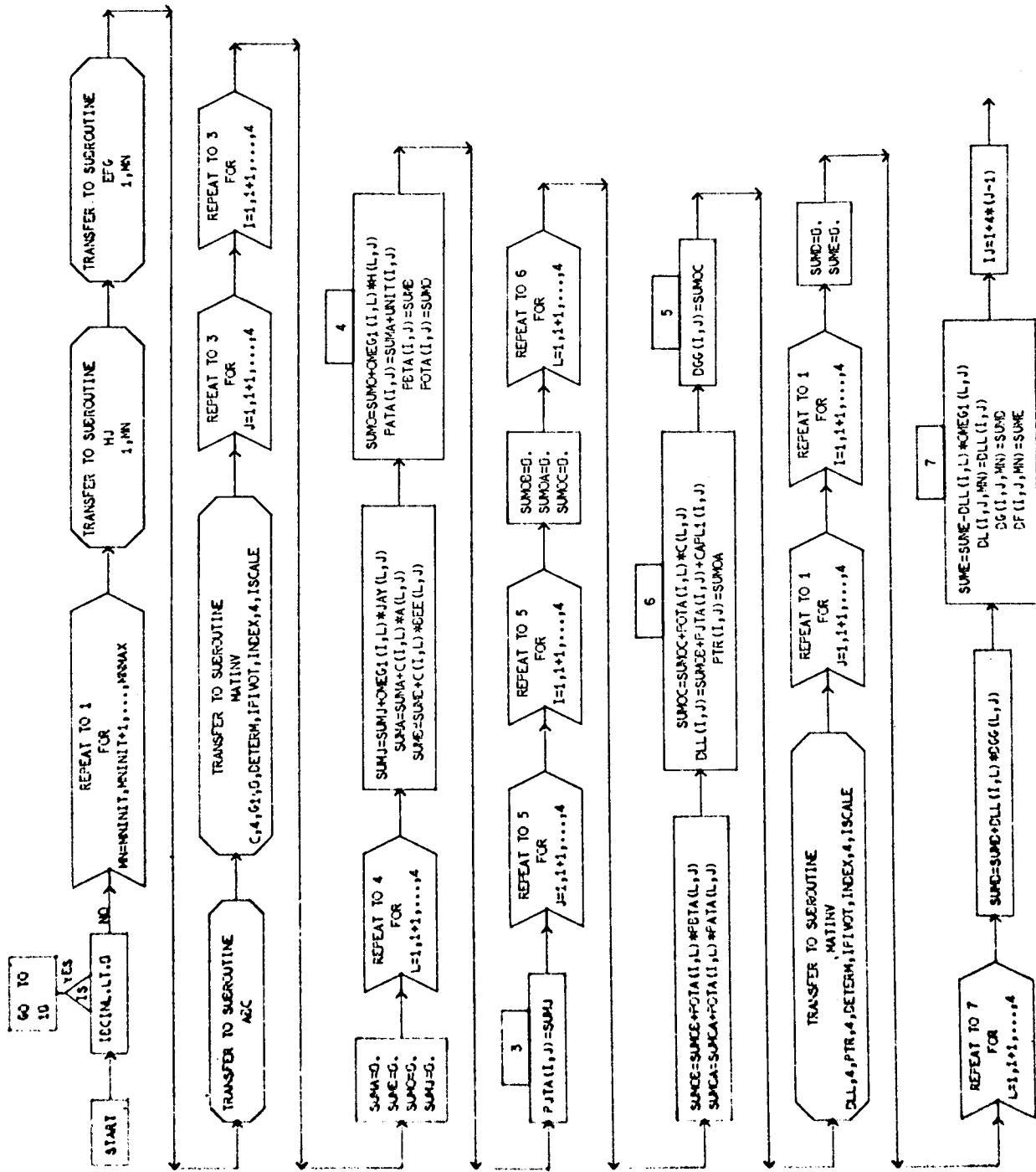


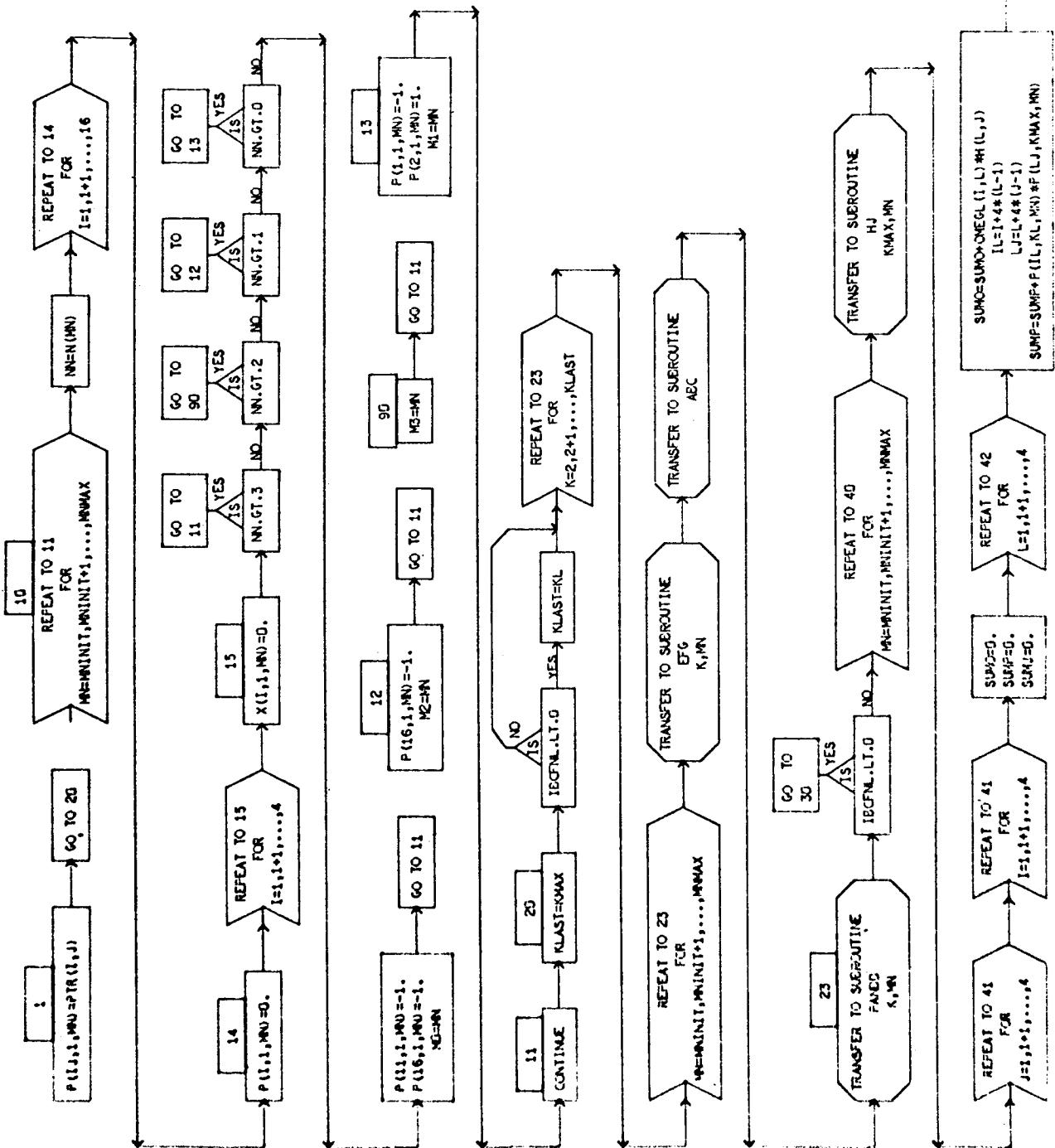


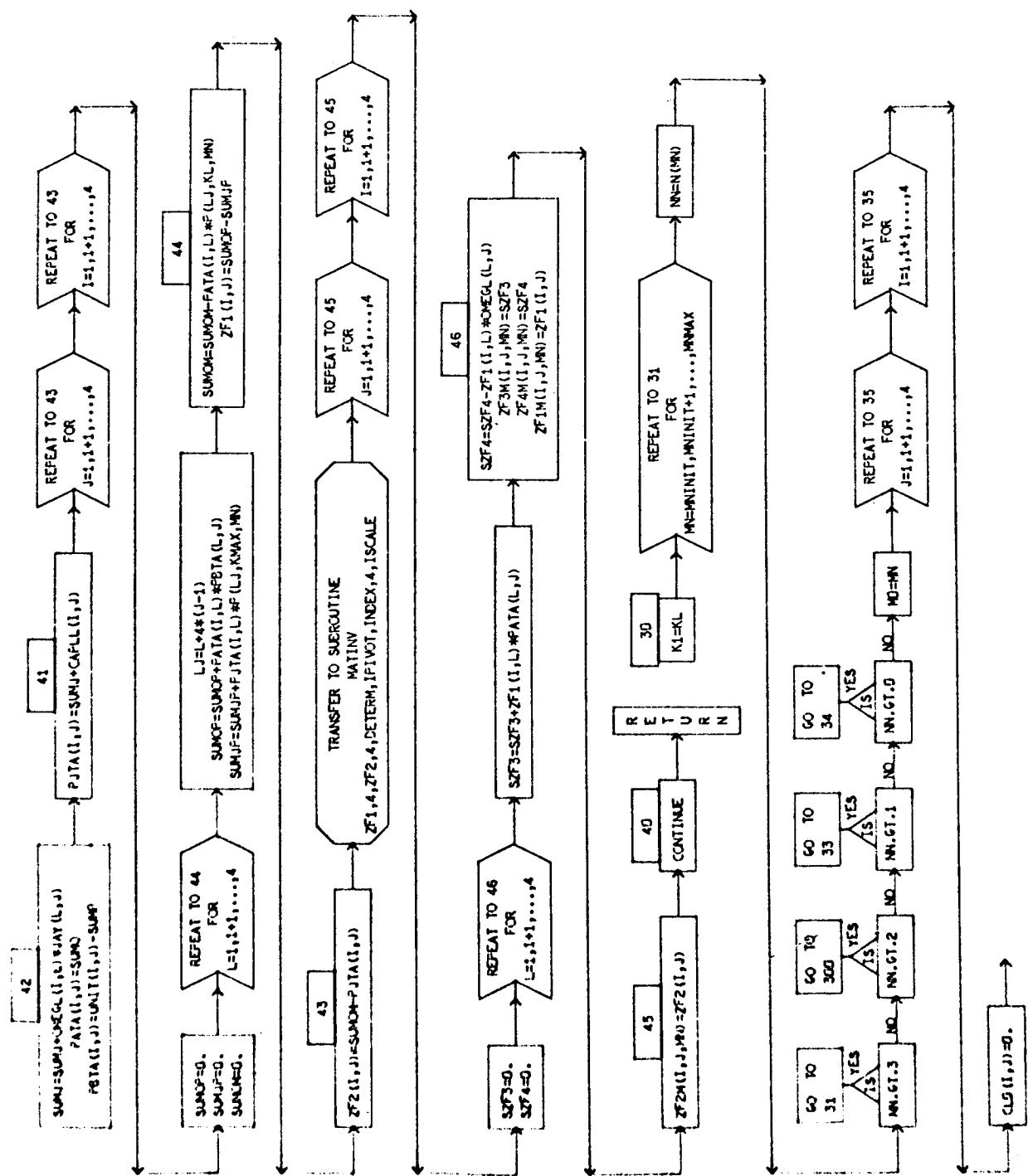
CIMENSIONED VARIABLES

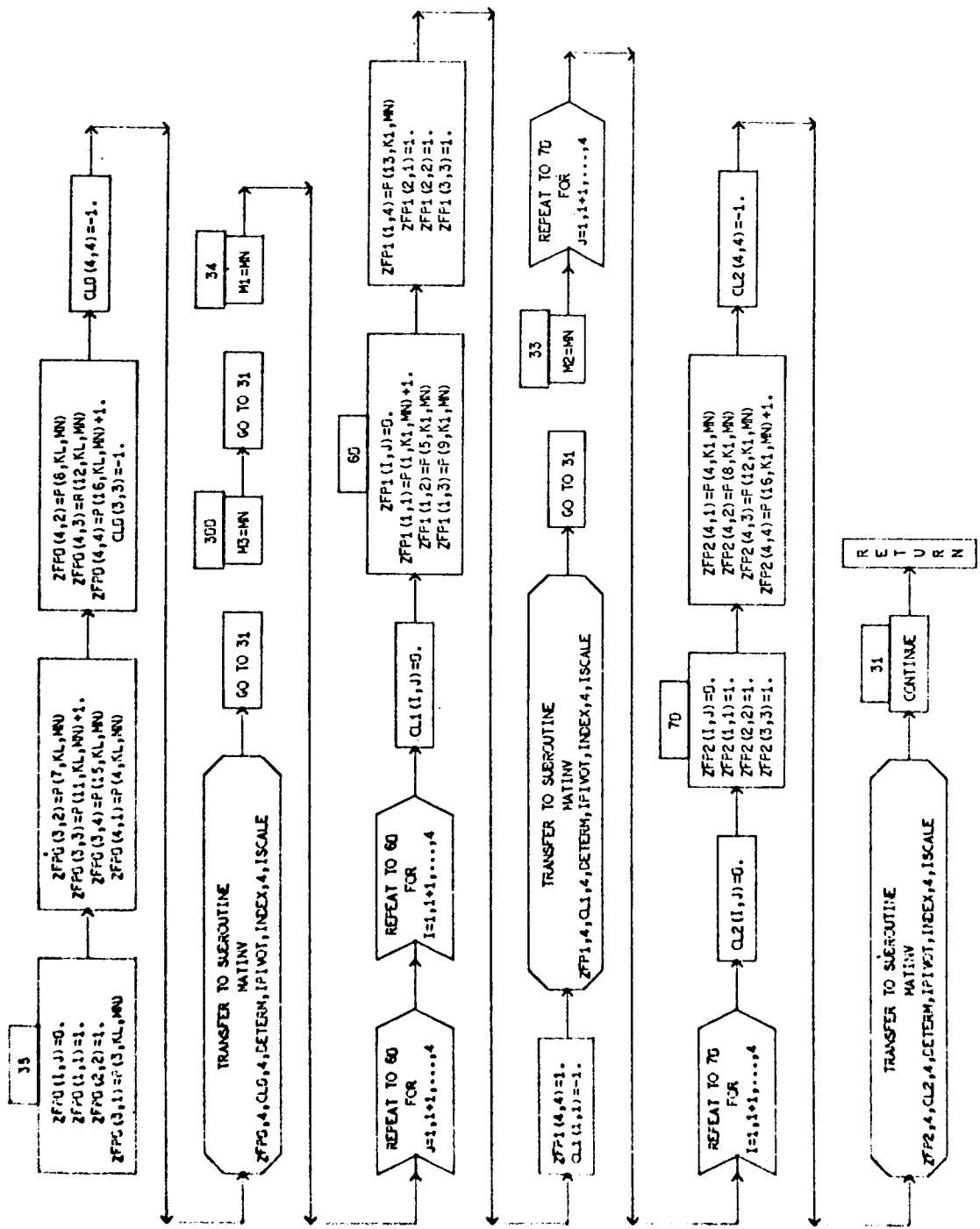
SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES
PITA	4,4	PITA	4,4	PITA	4,4	PITA	4,4
PRR	4,4	DG	4,4	ZF1	4,4	ZF2	4,4
ZP1	4,4	ZP2	4,4	IPIVOT	4	INDEX	4,2
Q1	4,4	Q2	4,4			CLD	4,4

SUBROUTINE MATRIX





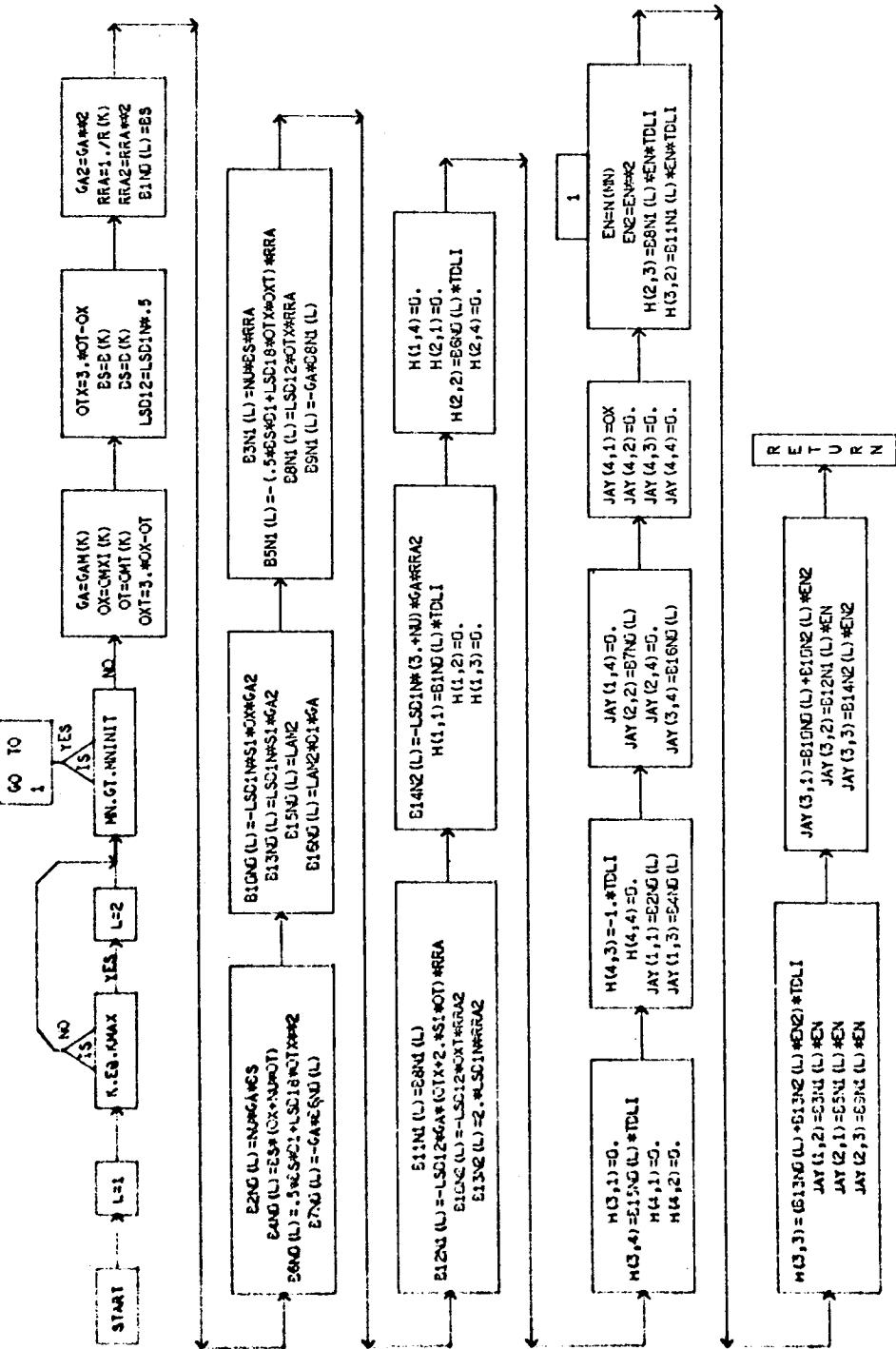




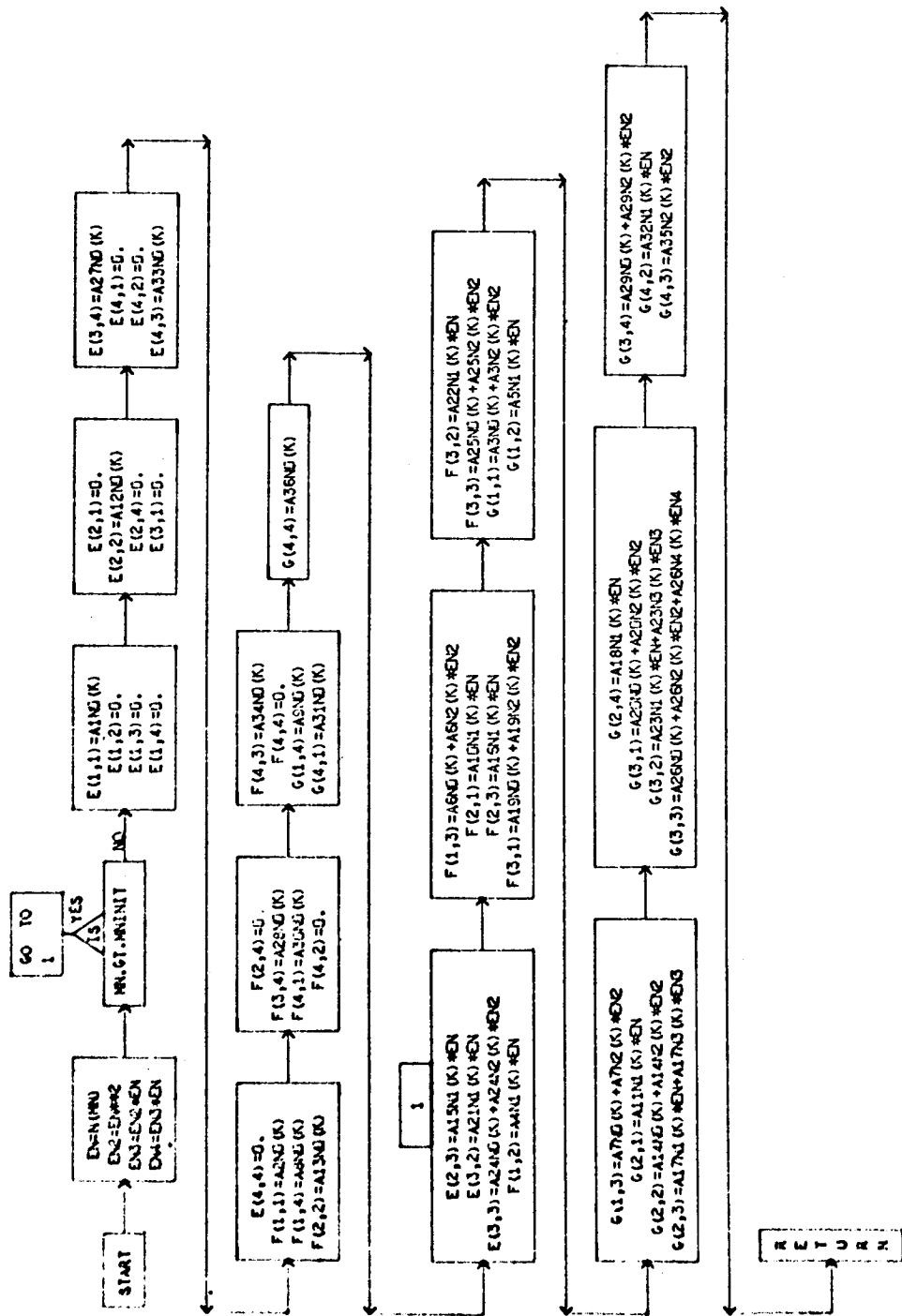
DIMENSIONED VARIABLES

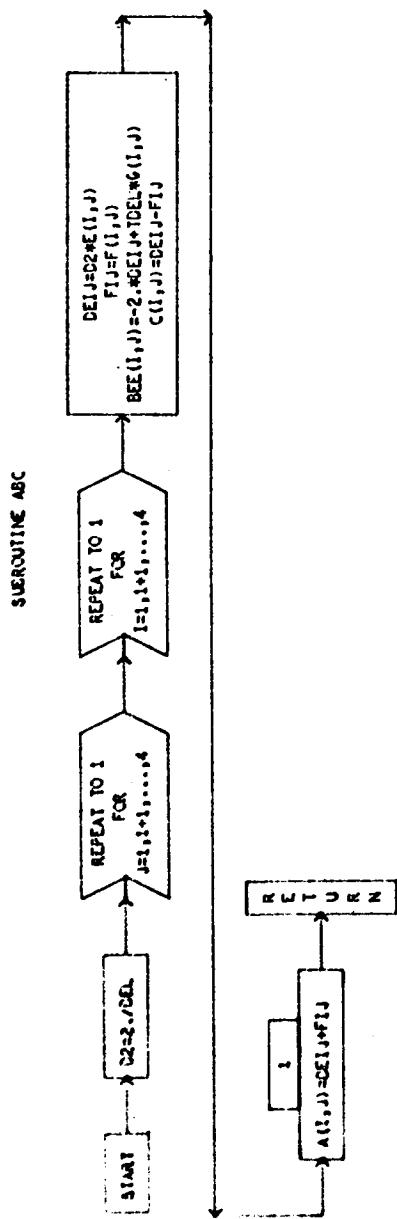
SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES
E140	2	E2ND	2	E4ND	2	E6ND	2
E14C	2	E3NC	2	E15G	2	E6G	2
E2ND	2	E8C	2	E9N	2	E11N	2
E14Z	2	E13N	2	E14Z	2	E12N	2

SUBROUTINE HJ (K, MN)



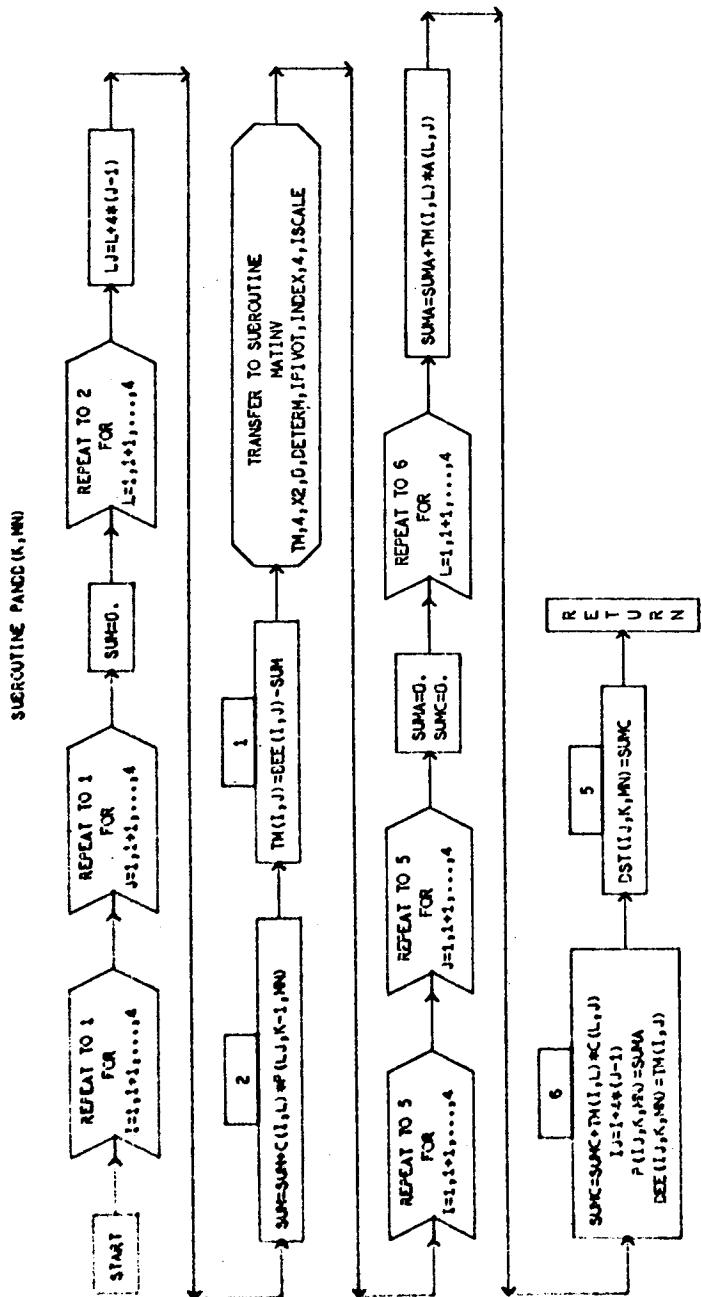
SUBROUTINE EFC(K, MN)





DIMENSIONED VARIABLES

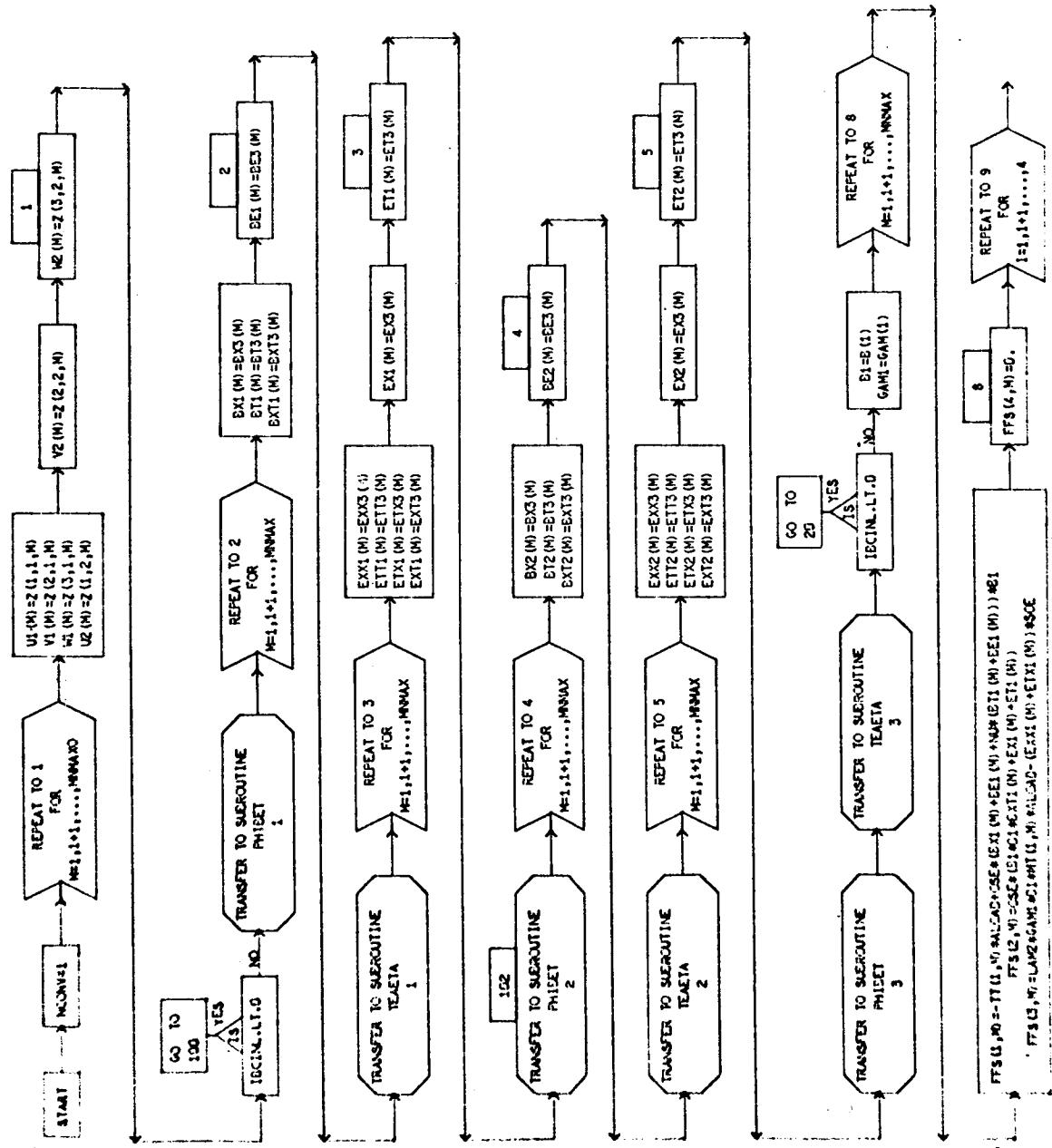
SYMOL	STORAGES	SYMOL	STORAGES	SYMOL	STORAGES	SYMOL	STORAGES
W	4,4	PIVOT	4	INDEX	4,2		

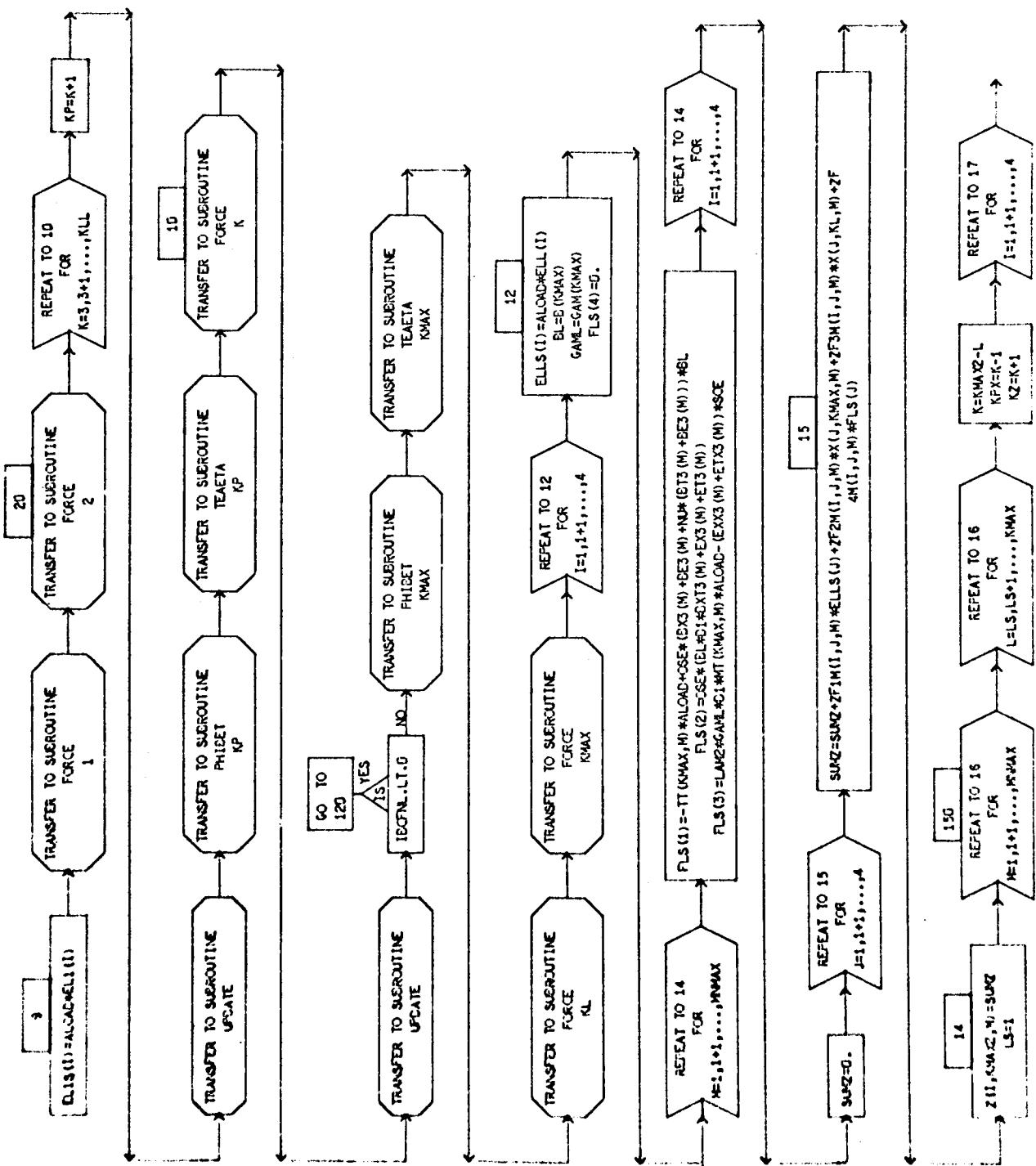


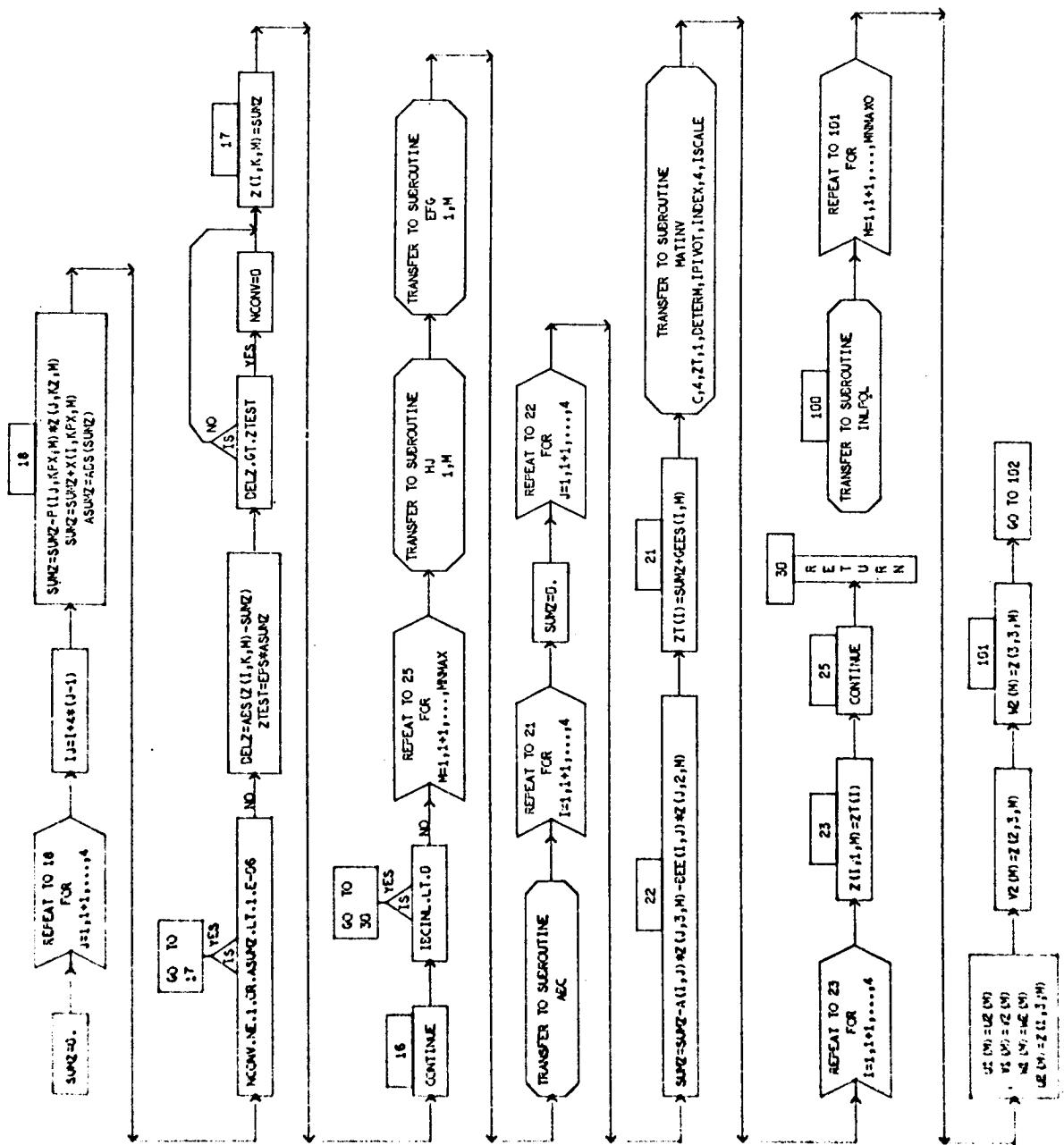
DIMENSIONED VARIABLES

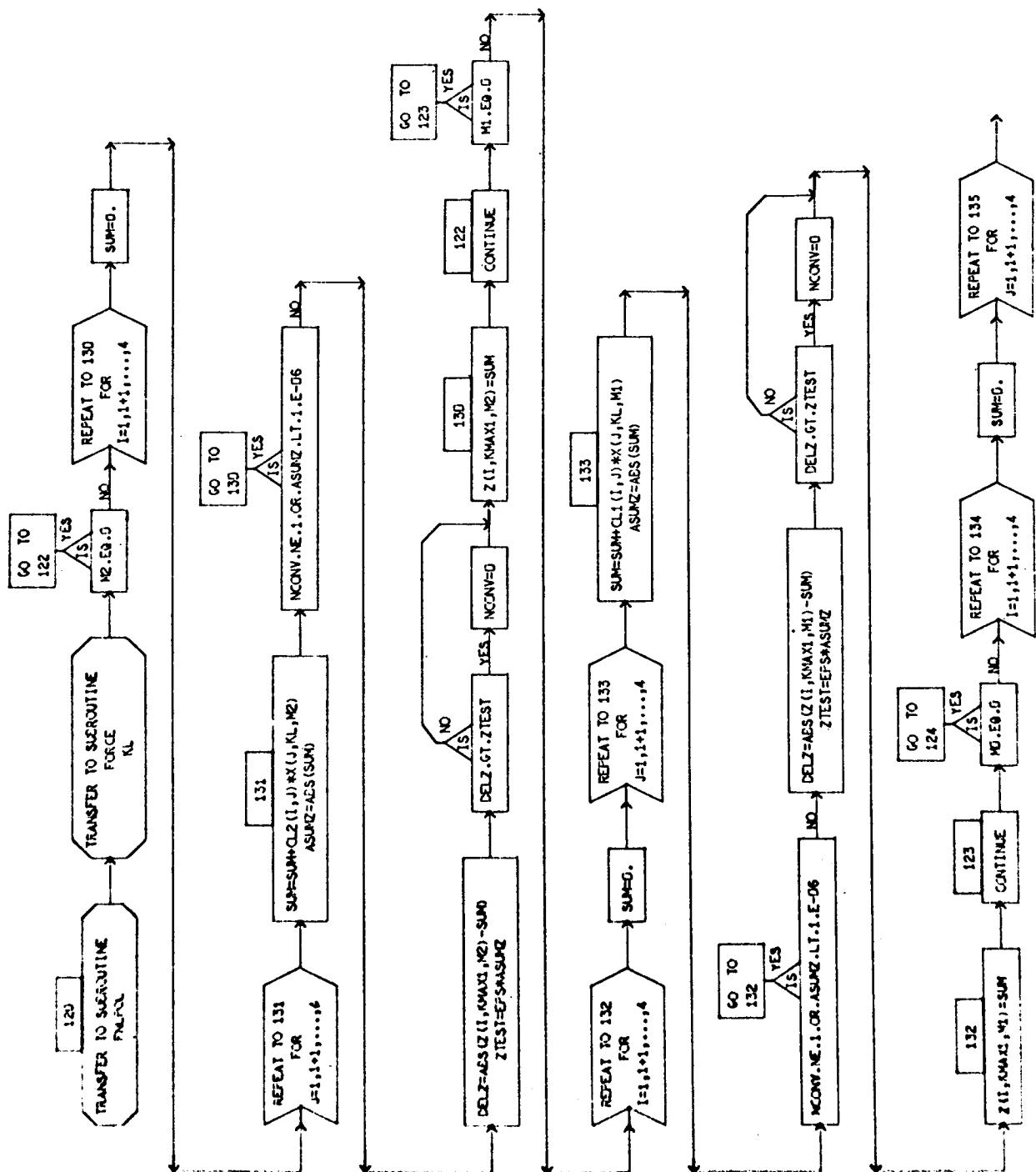
SOURCE	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES	SYMBOL	STORAGES
RHS	4	RHS	4	Z1	4	PIVOT	4	INDEX	4,2
CQ	4,4	CQ1	4,4	CQ2	4,4				

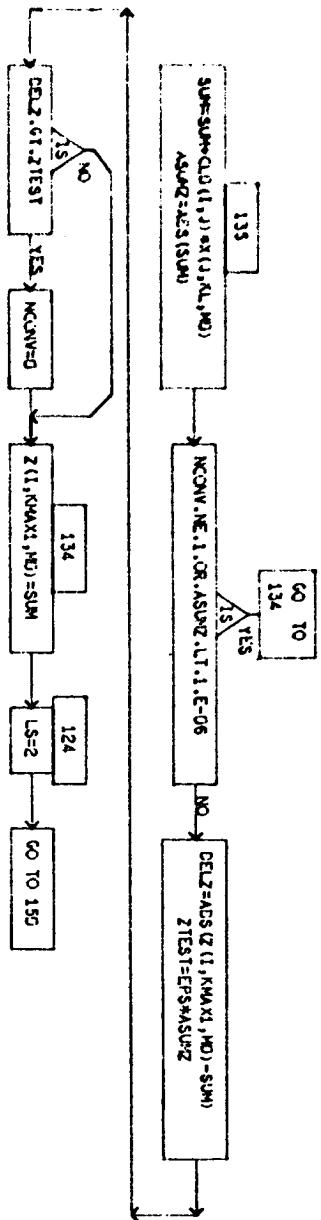
SUBROUTINE XANQZ

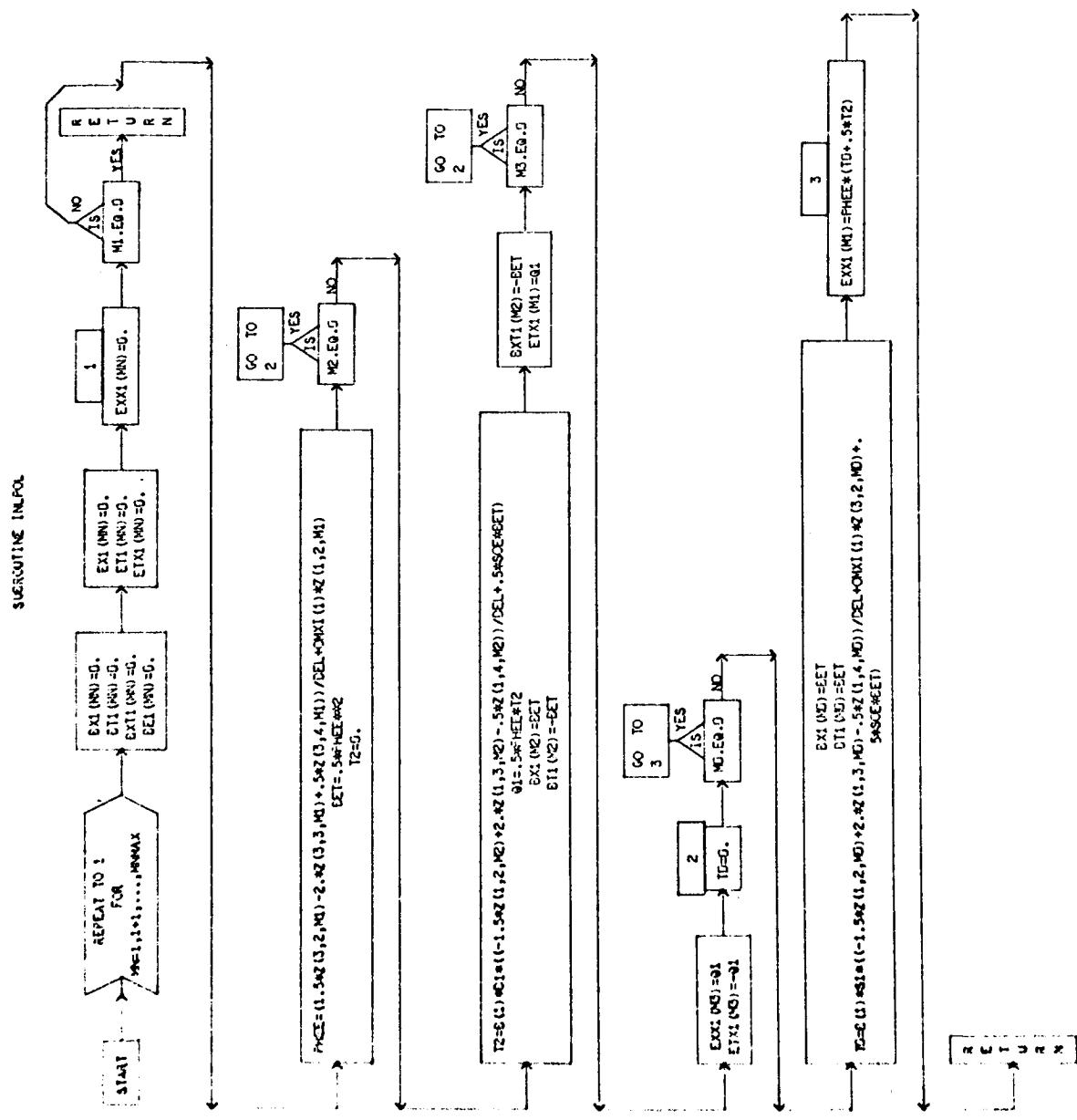




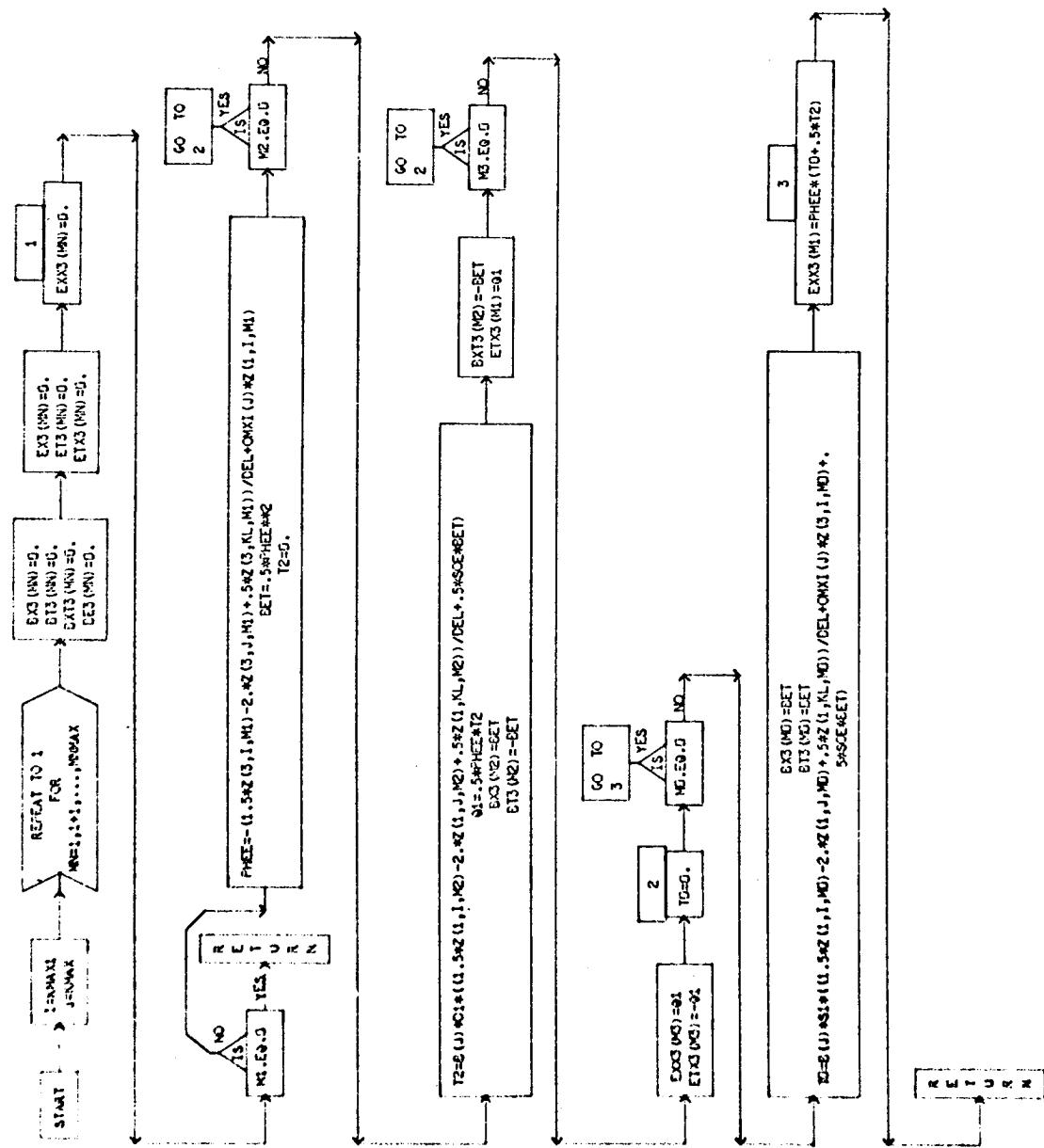




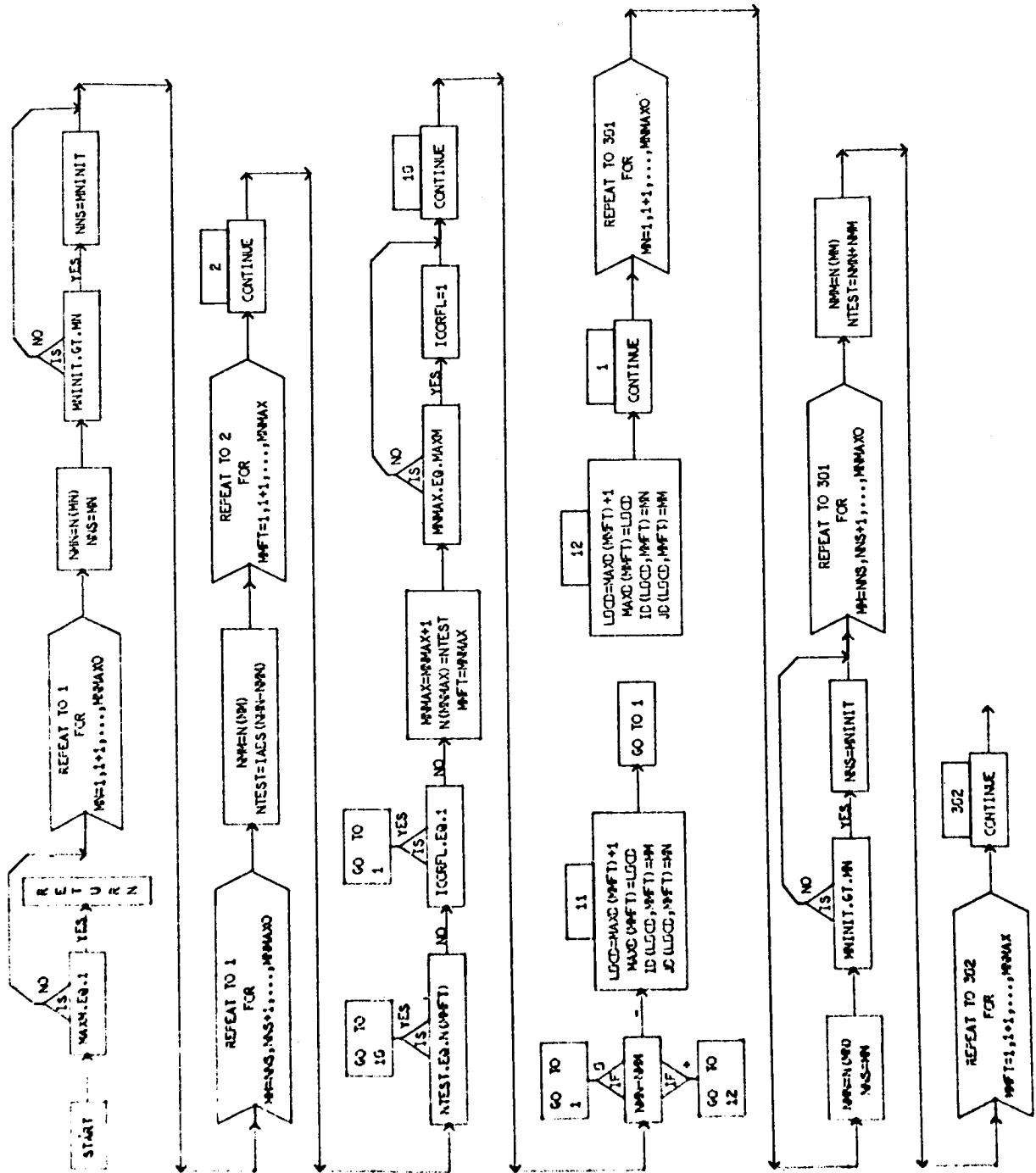


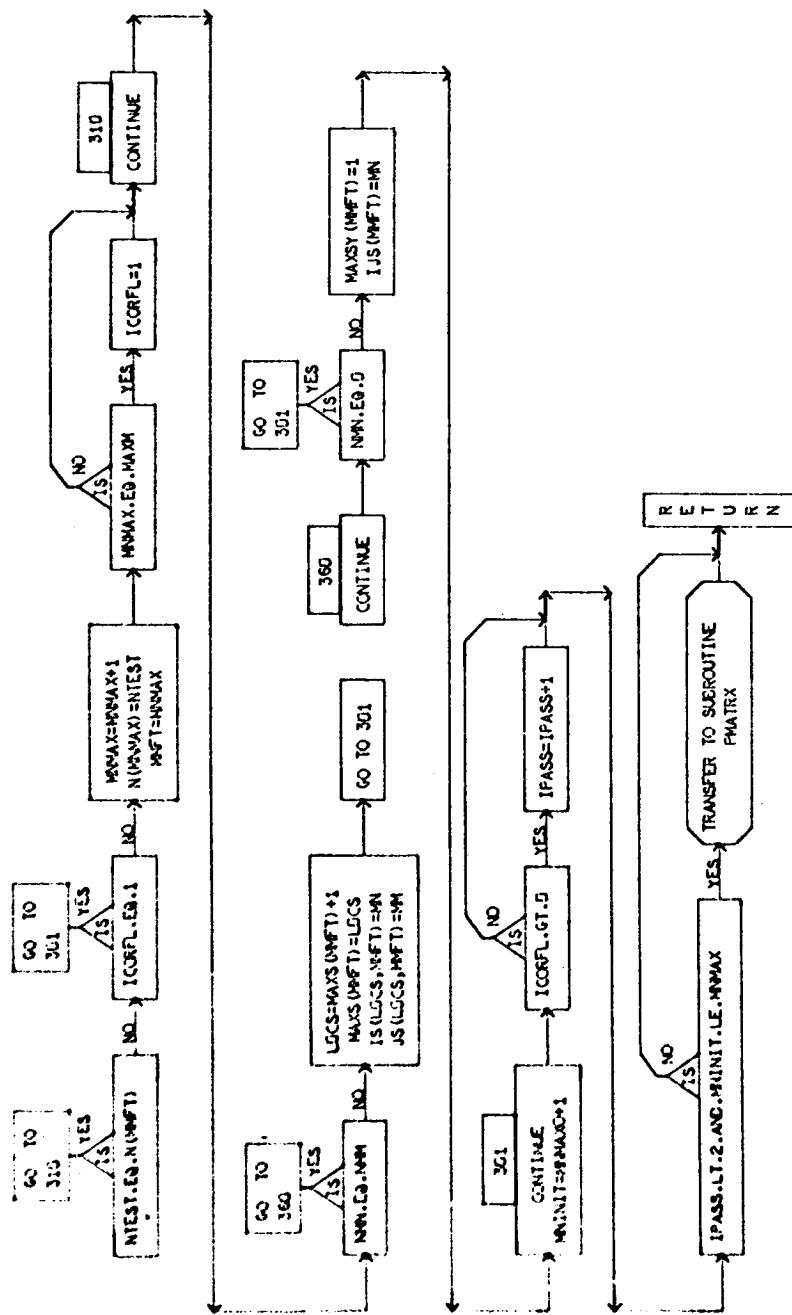


SUBROUTINE FINFLQ



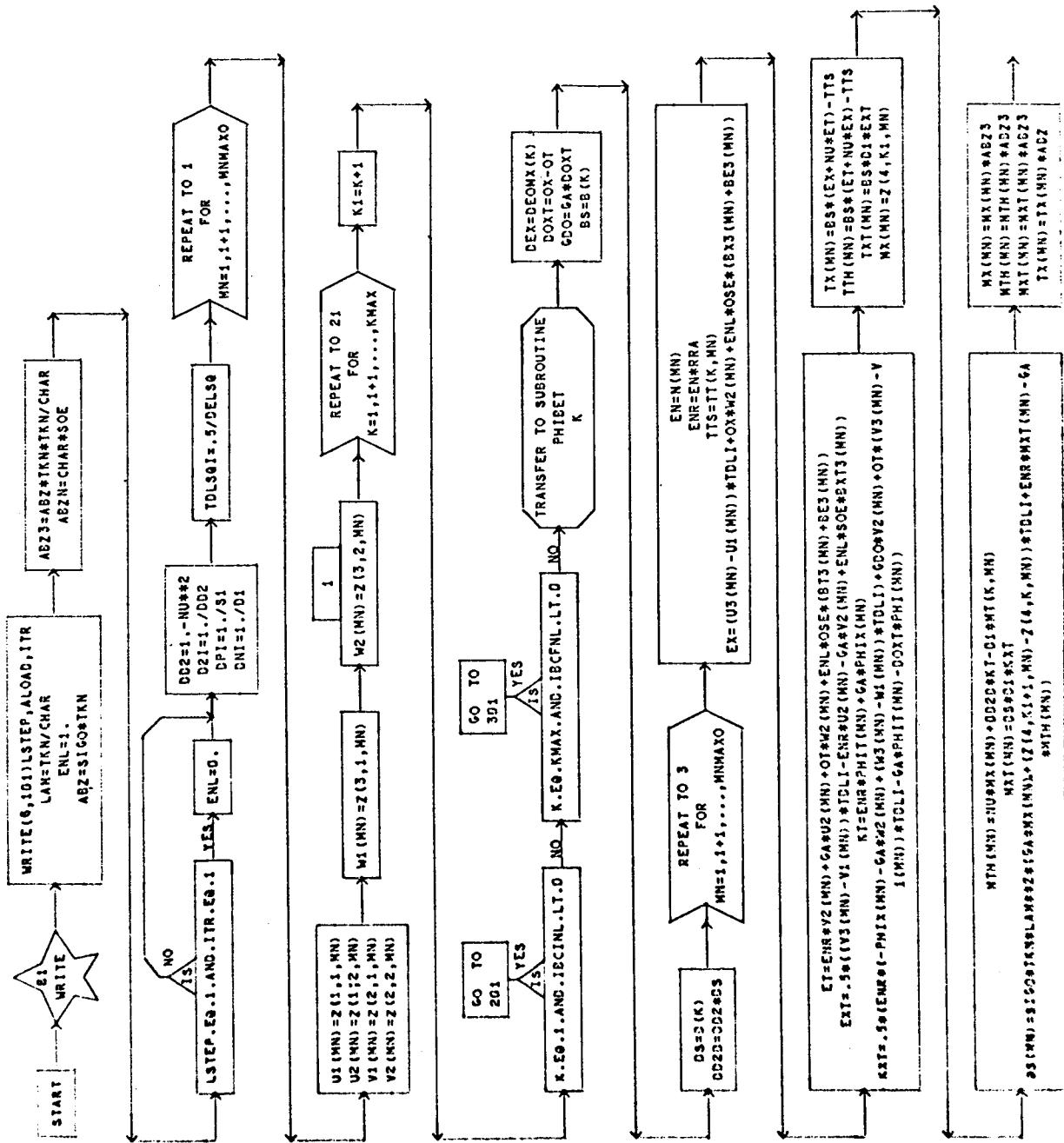
SUBROUTINE MACES

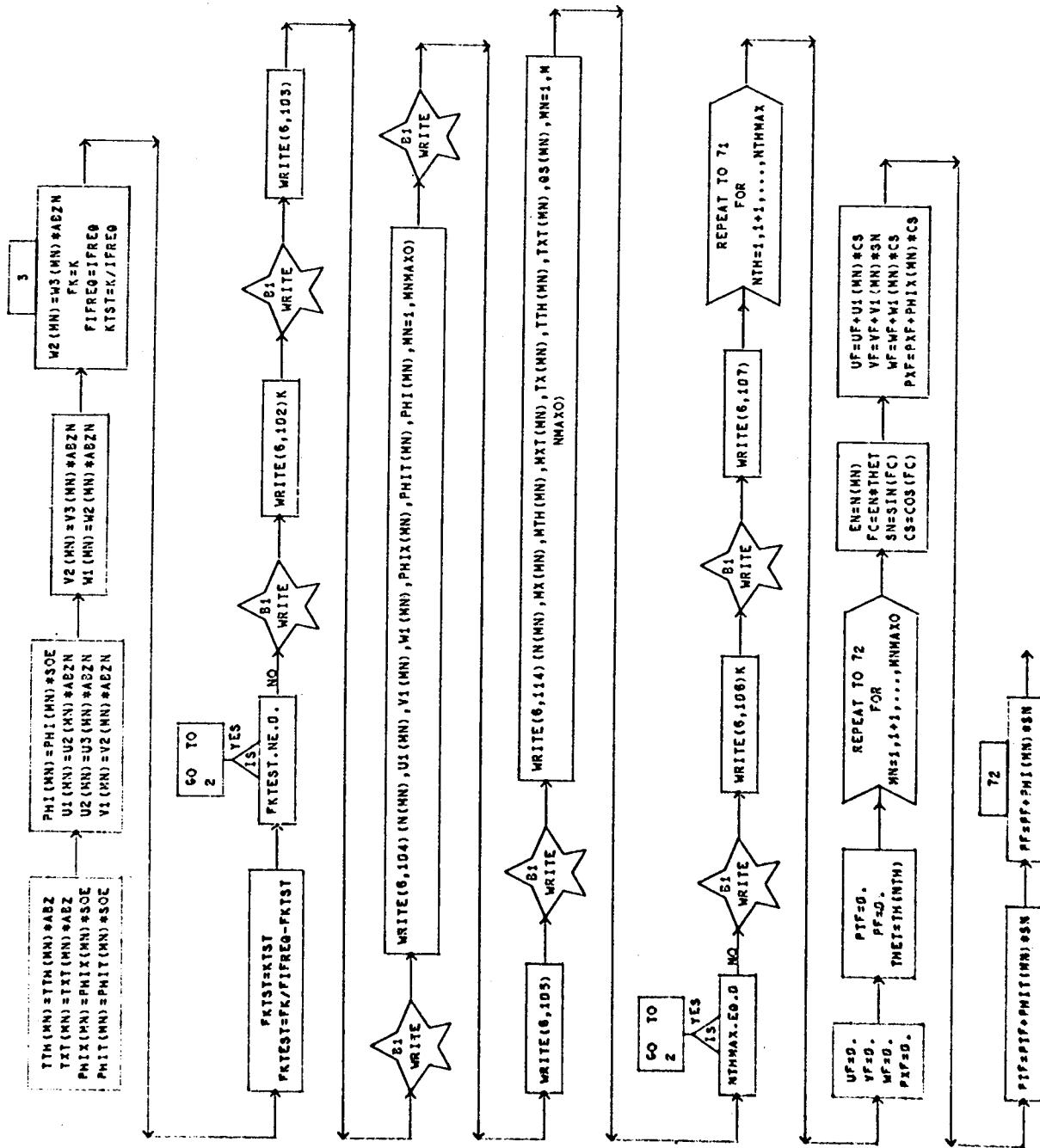


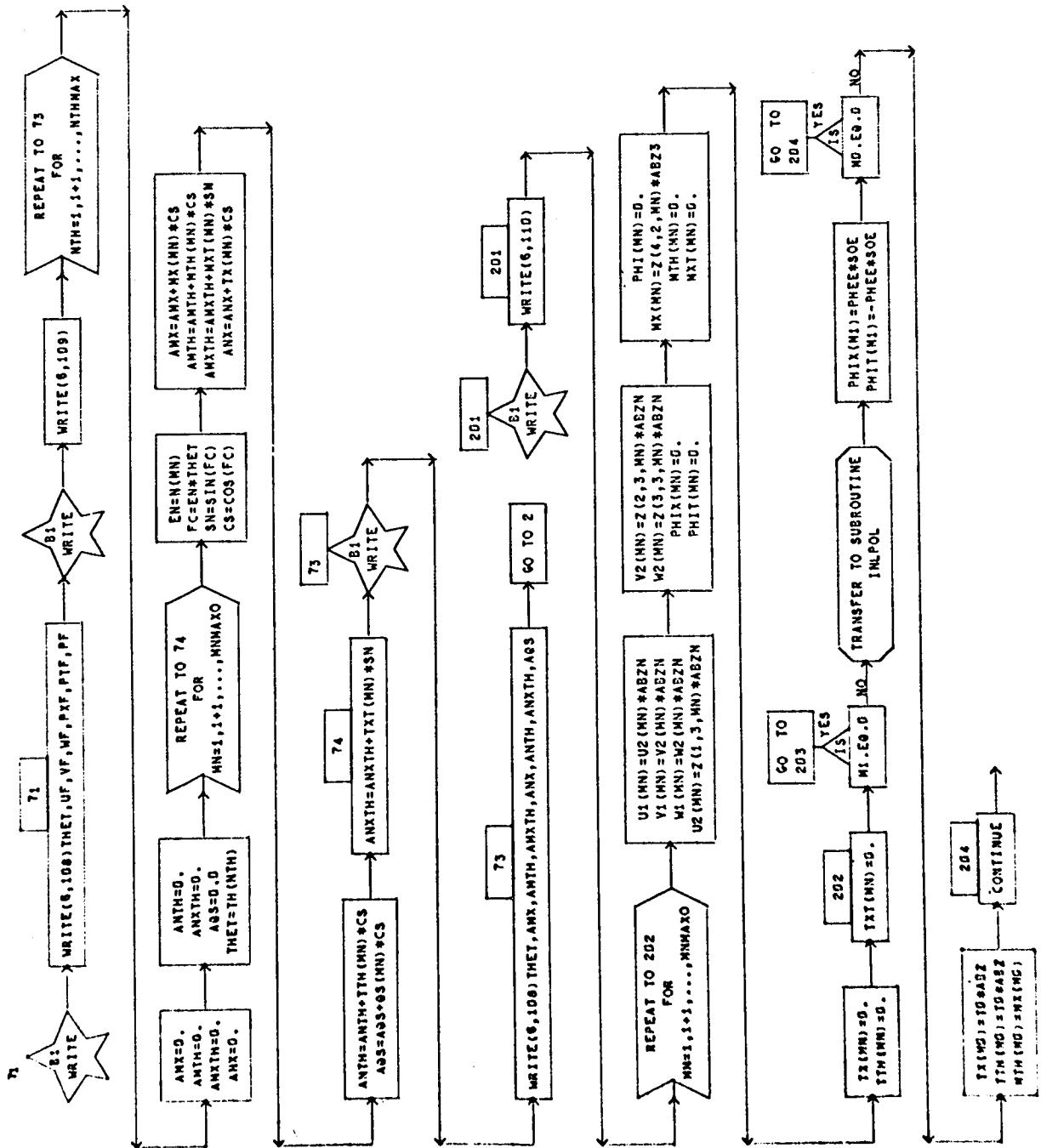


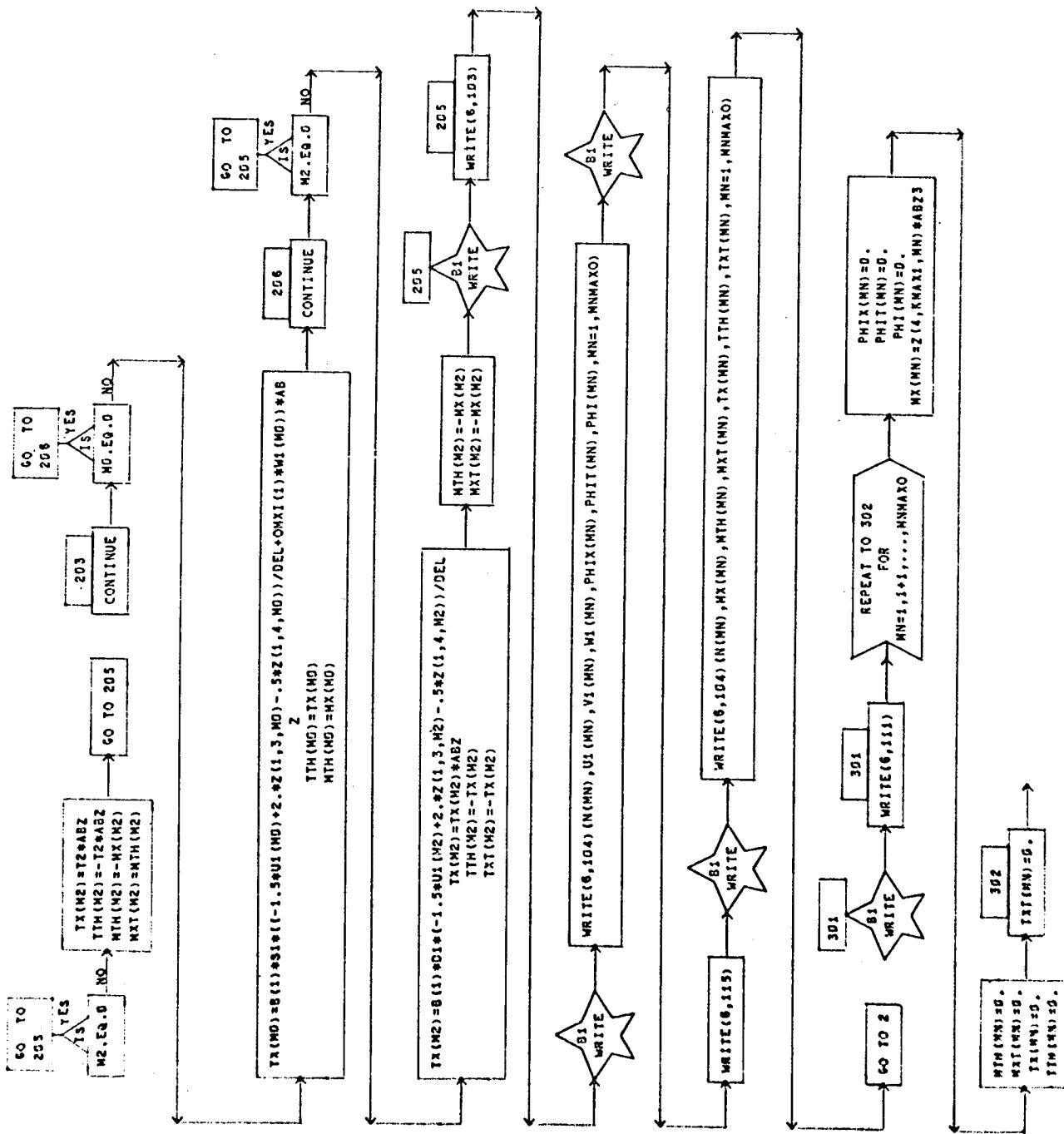
DIMENSIONED VARIABLES

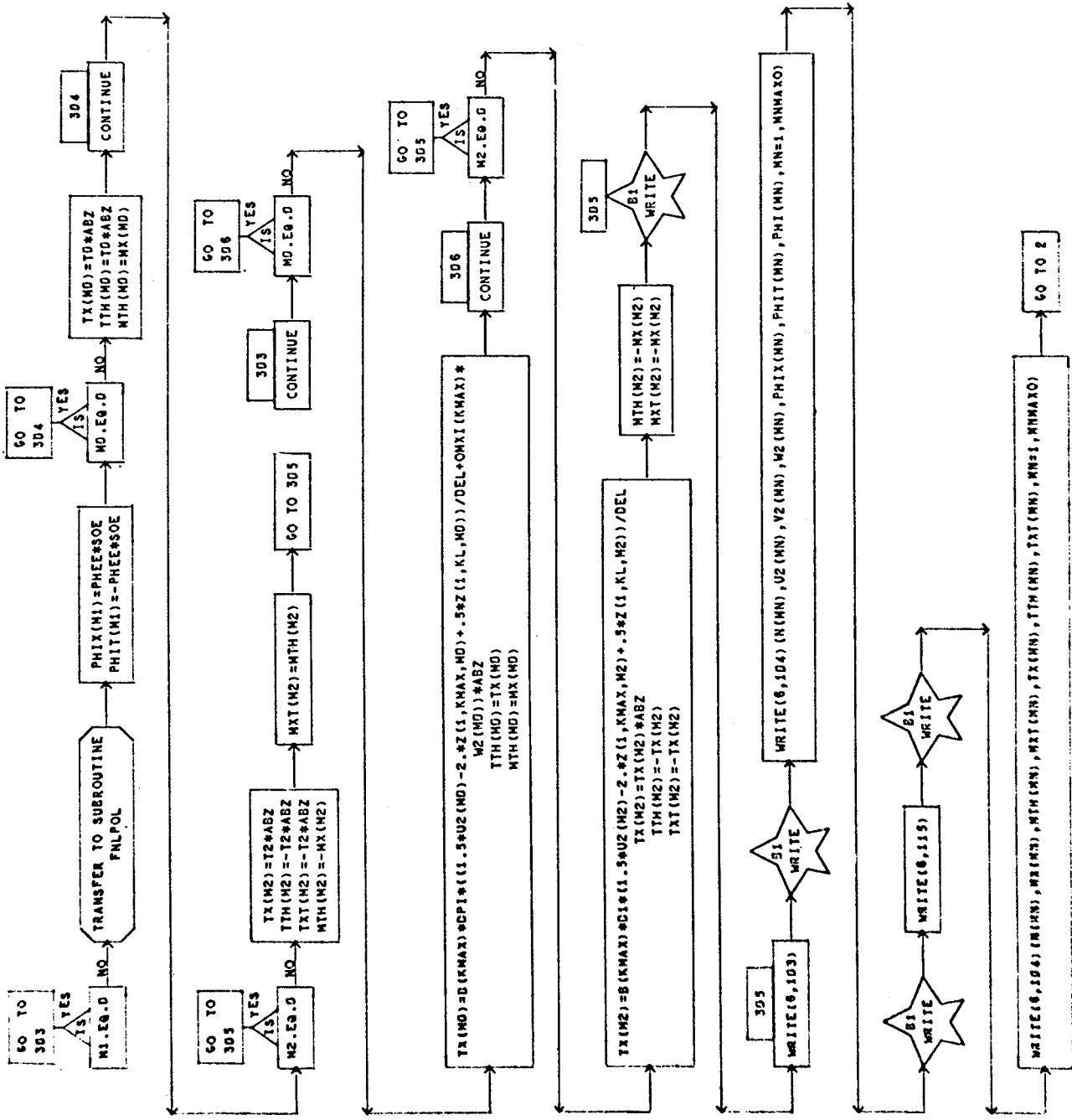
SYMBOL	STORAGES								
TX	10	TH	10	TXT	10	WT	10	WT	10
MX	10	QS	10						

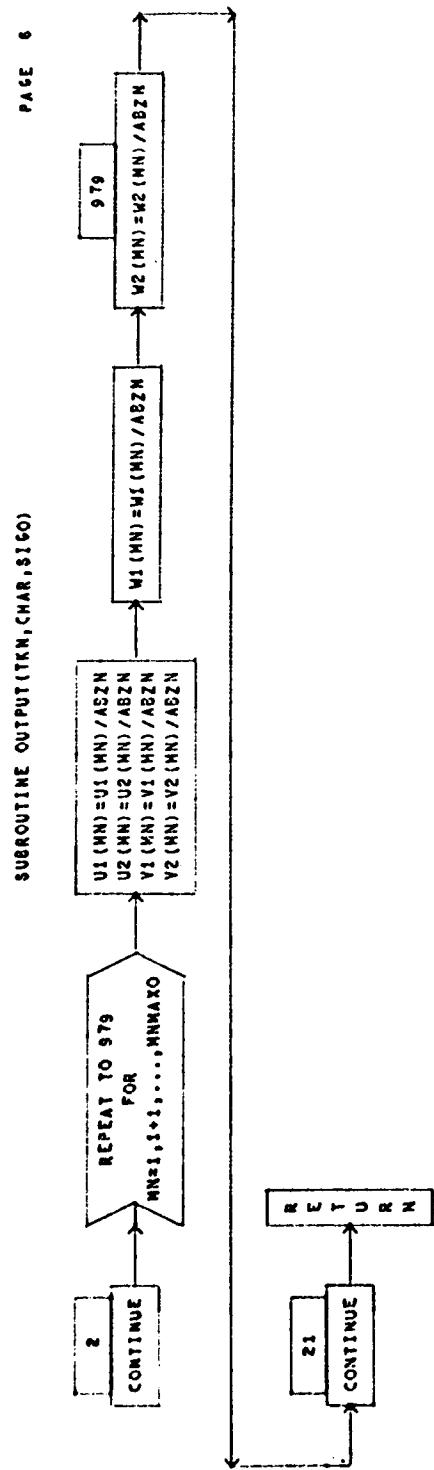




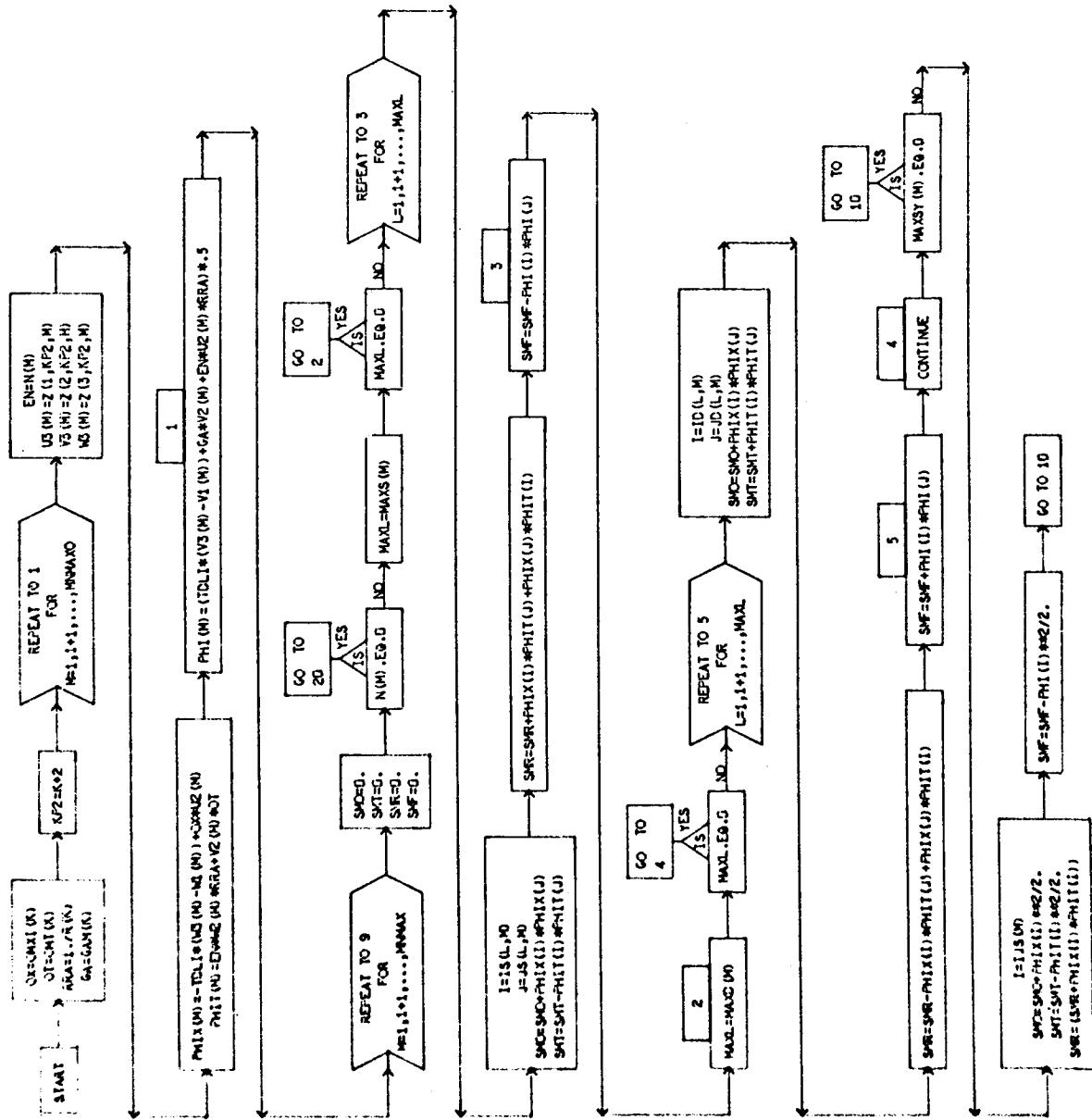


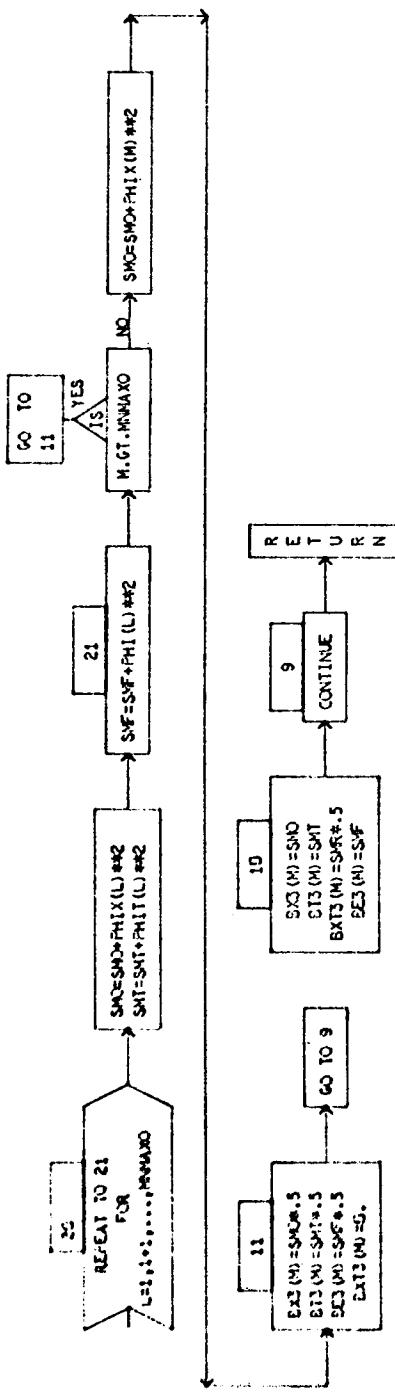






SUBROUTINE PHICET(M)

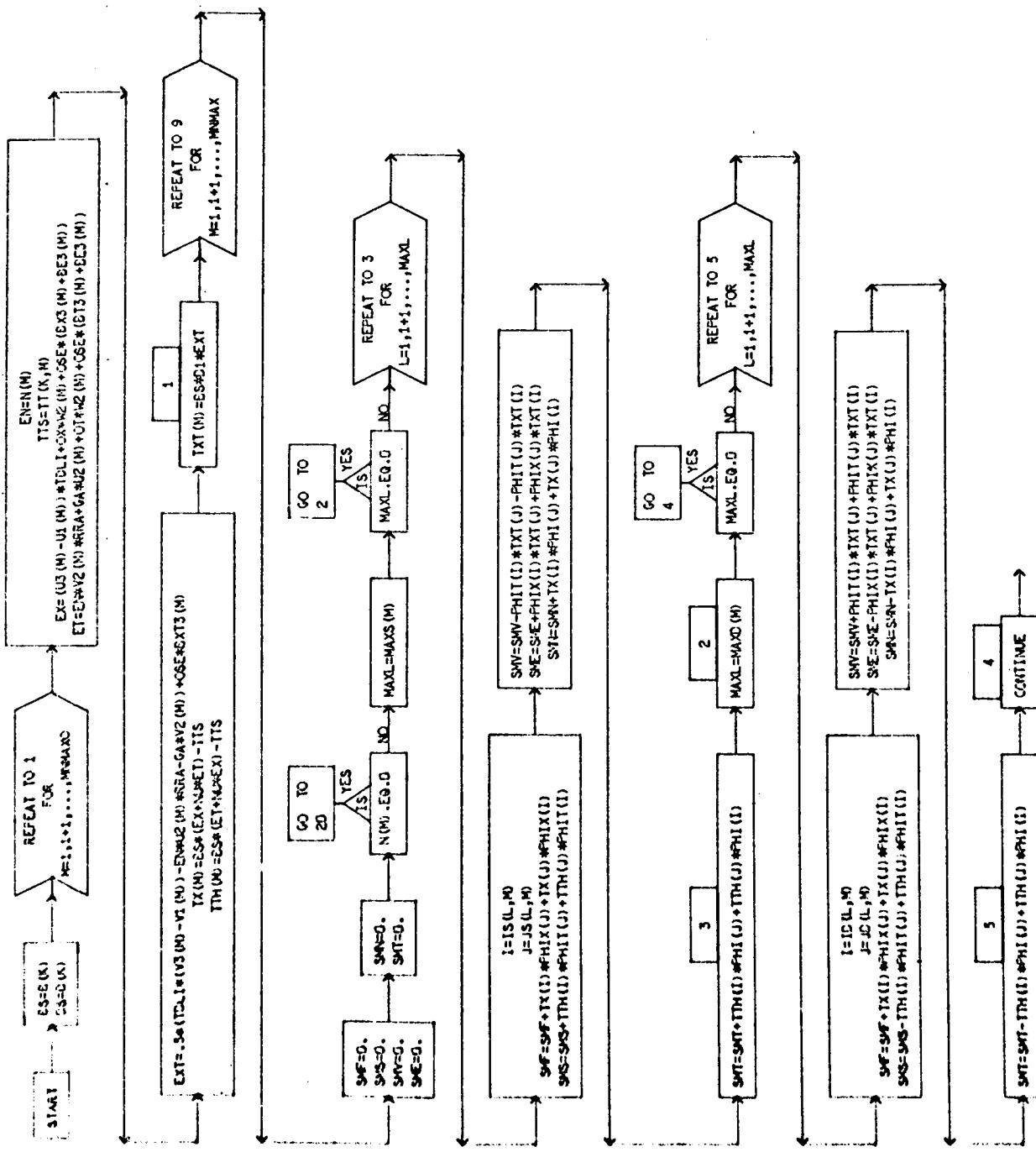


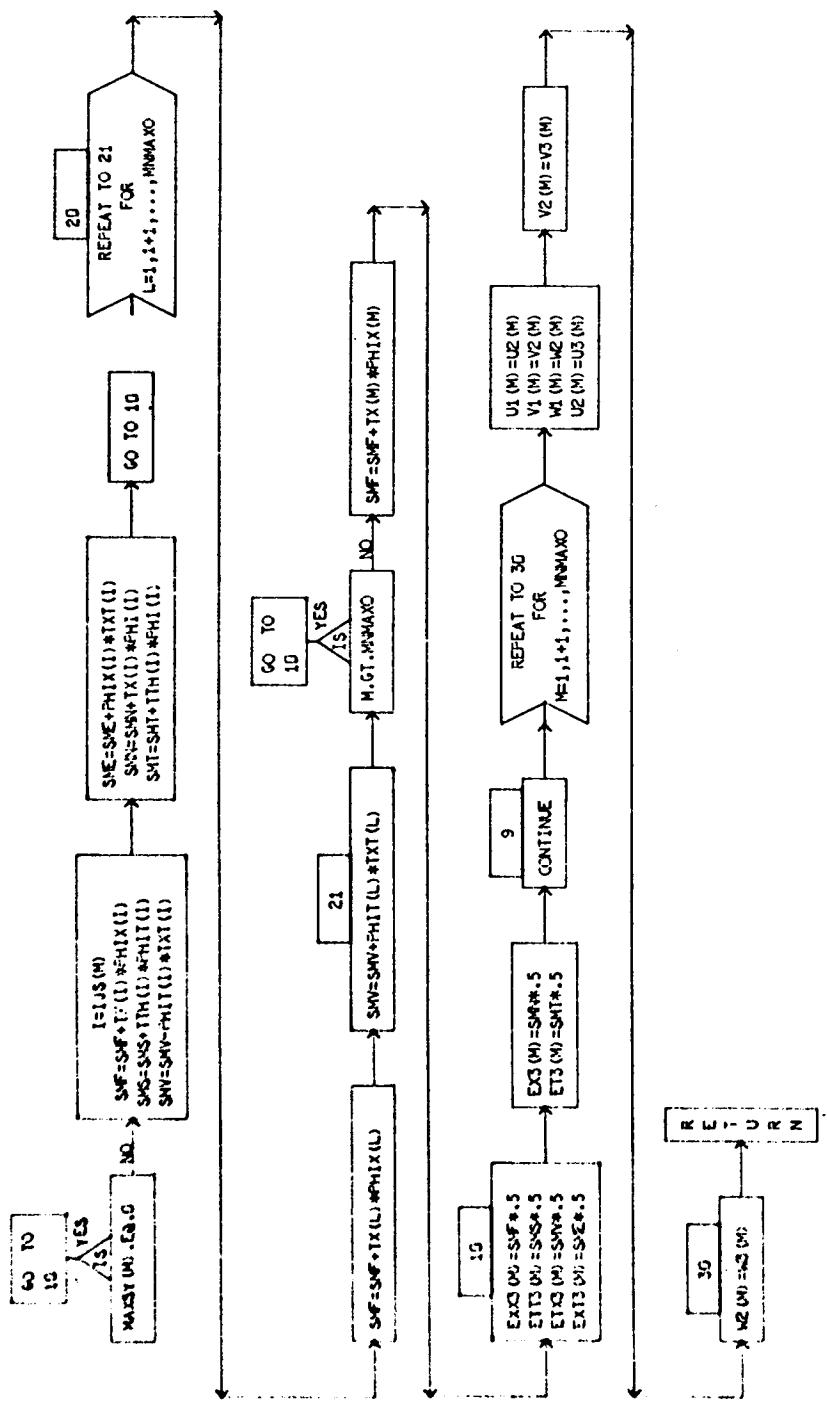


DIMENSIONED VARIABLES

SYMBOL	STORAGES								
TR	12	TM	1G	TXT	1G				

SUBROUTINE TEKETIA (M)

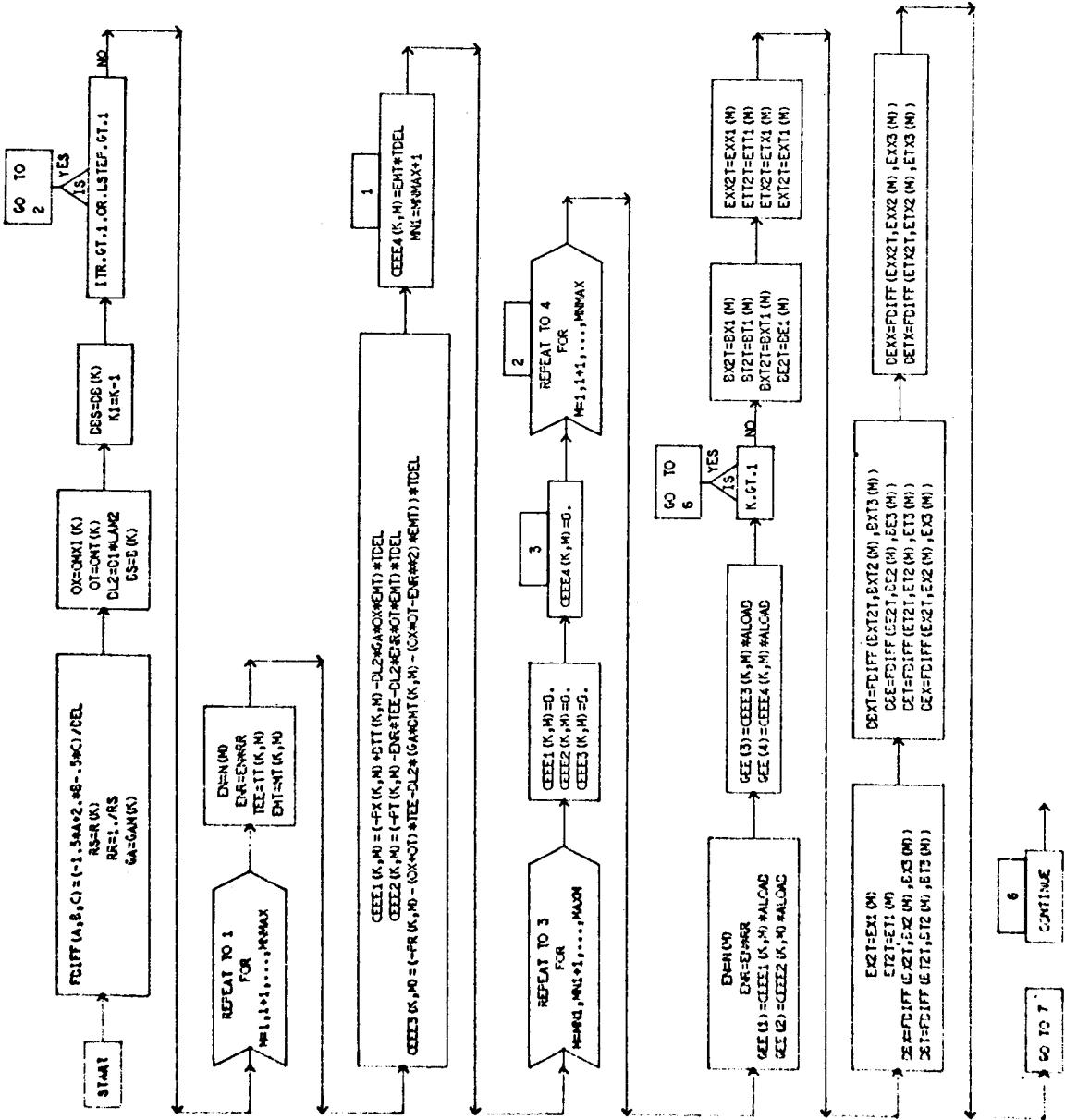


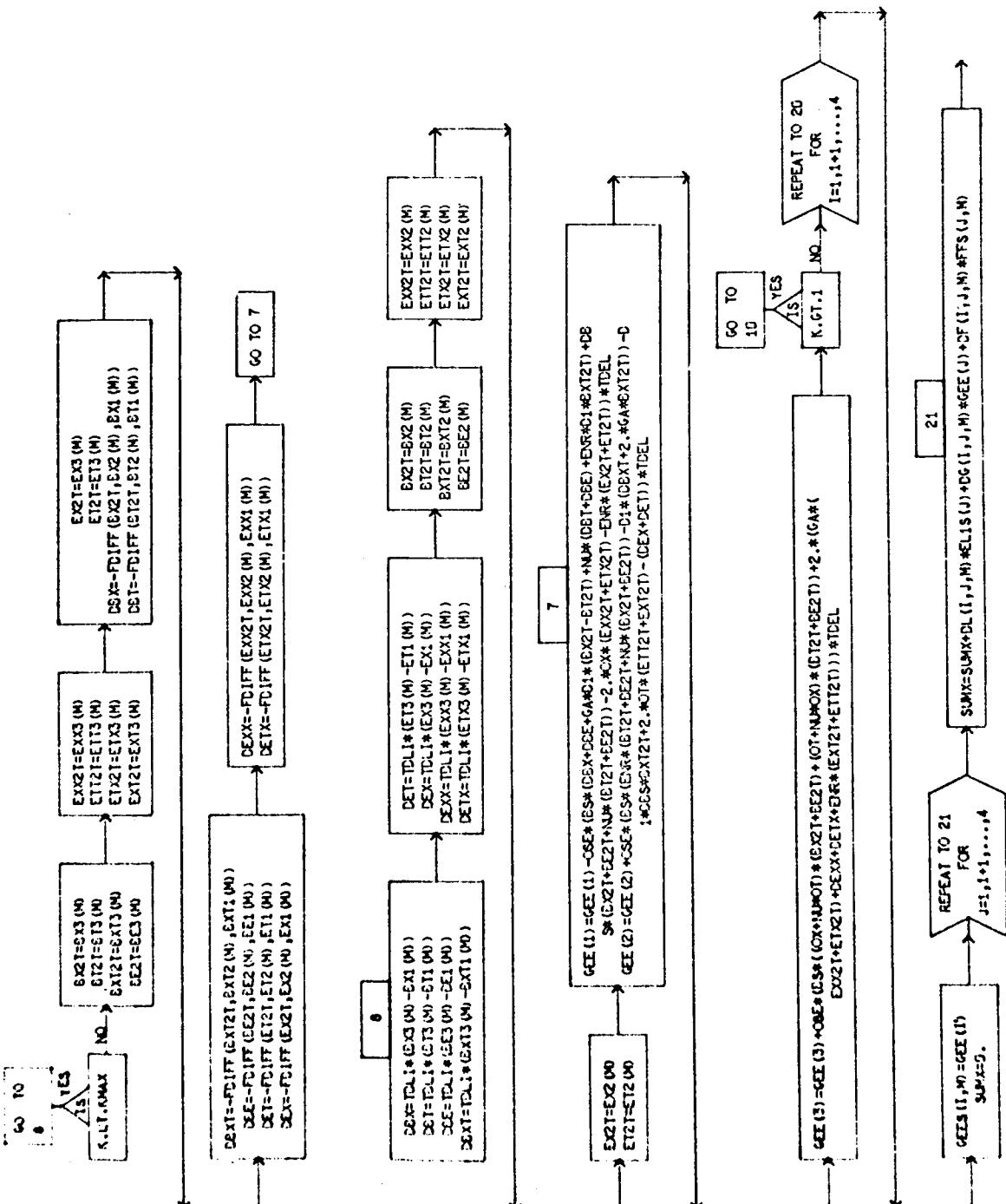


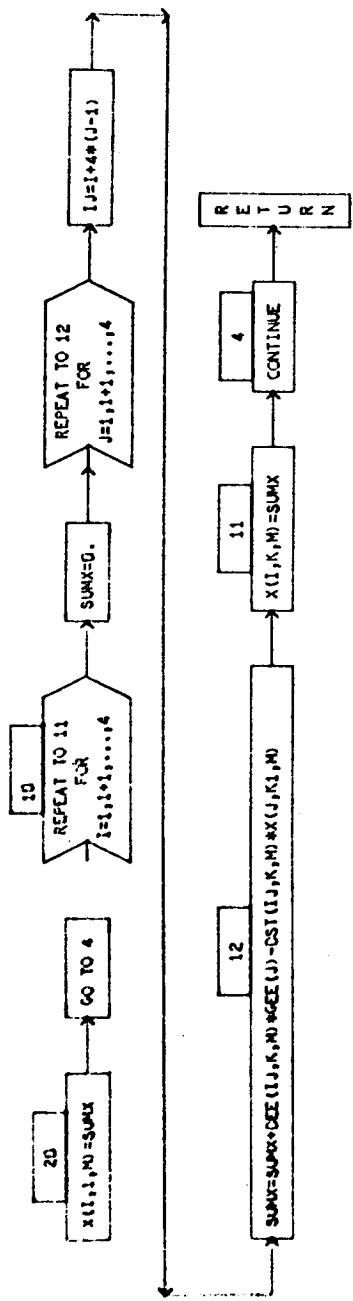
C I M E N S I O N E C V A R I A B L E S

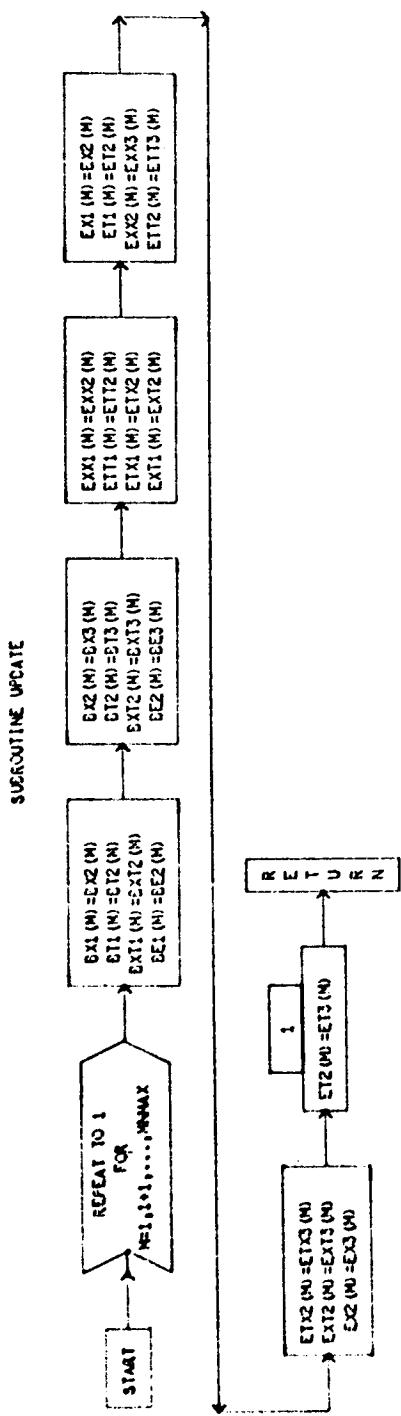
S Y M B O L	S T O R A G E S	S Y M B O L	S T O R A G E S	S Y M B O L	S T O R A G E S	S Y M B O L	S T O R A G E S
CEE	4	CEE1	20,10	CEE2	20,10	CEE3	20,10

SUBROUTINE FORCE(M)



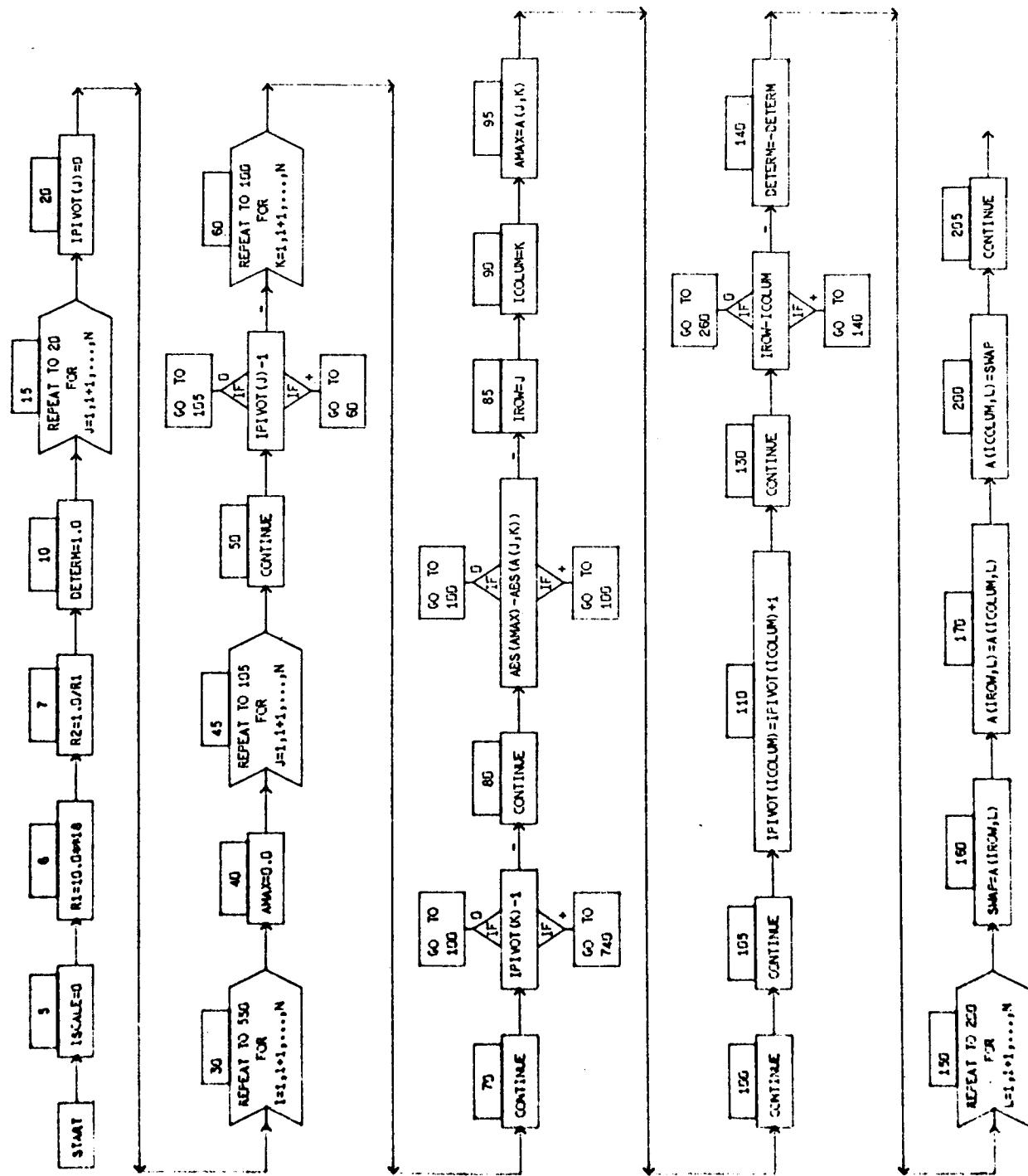


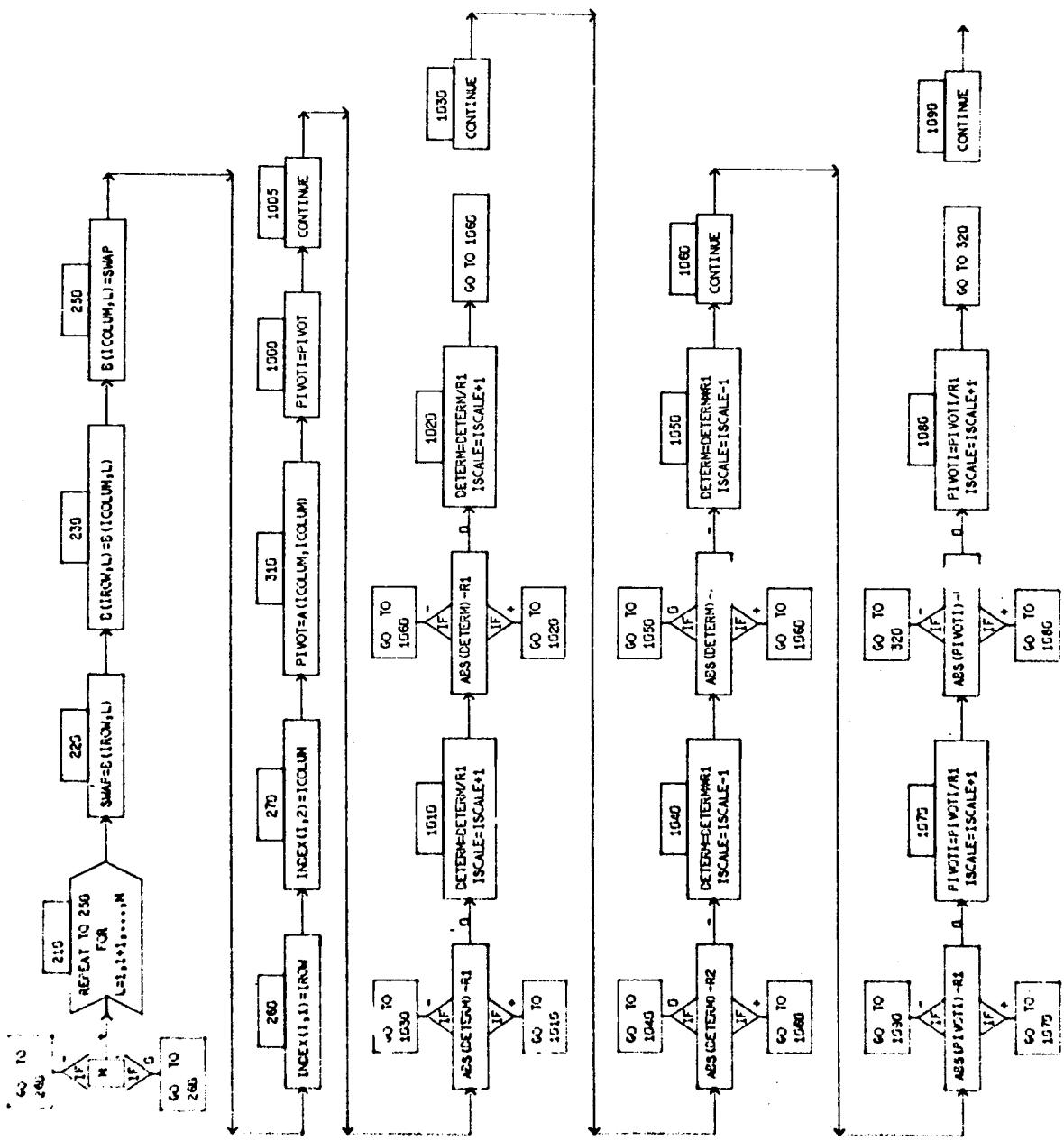


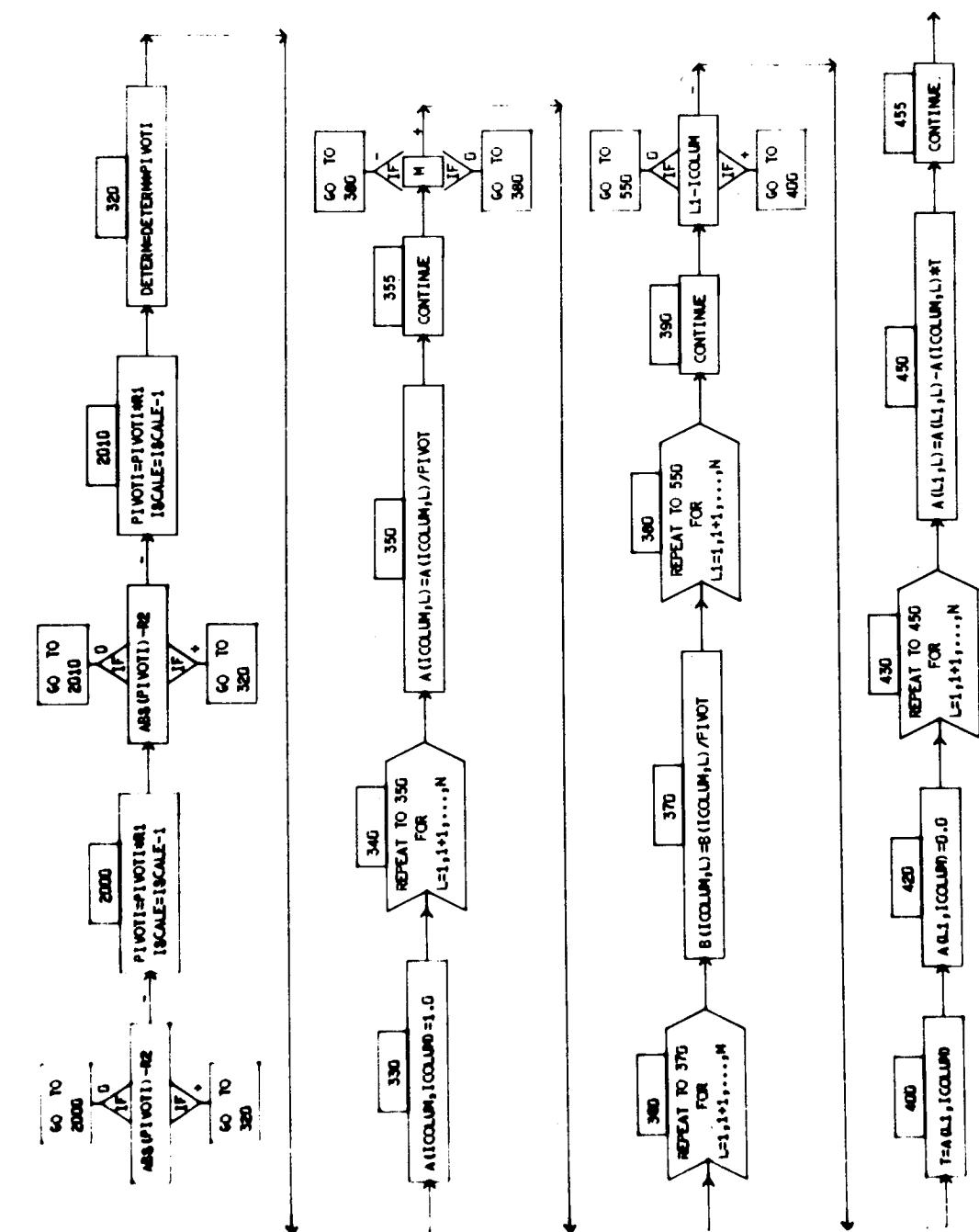


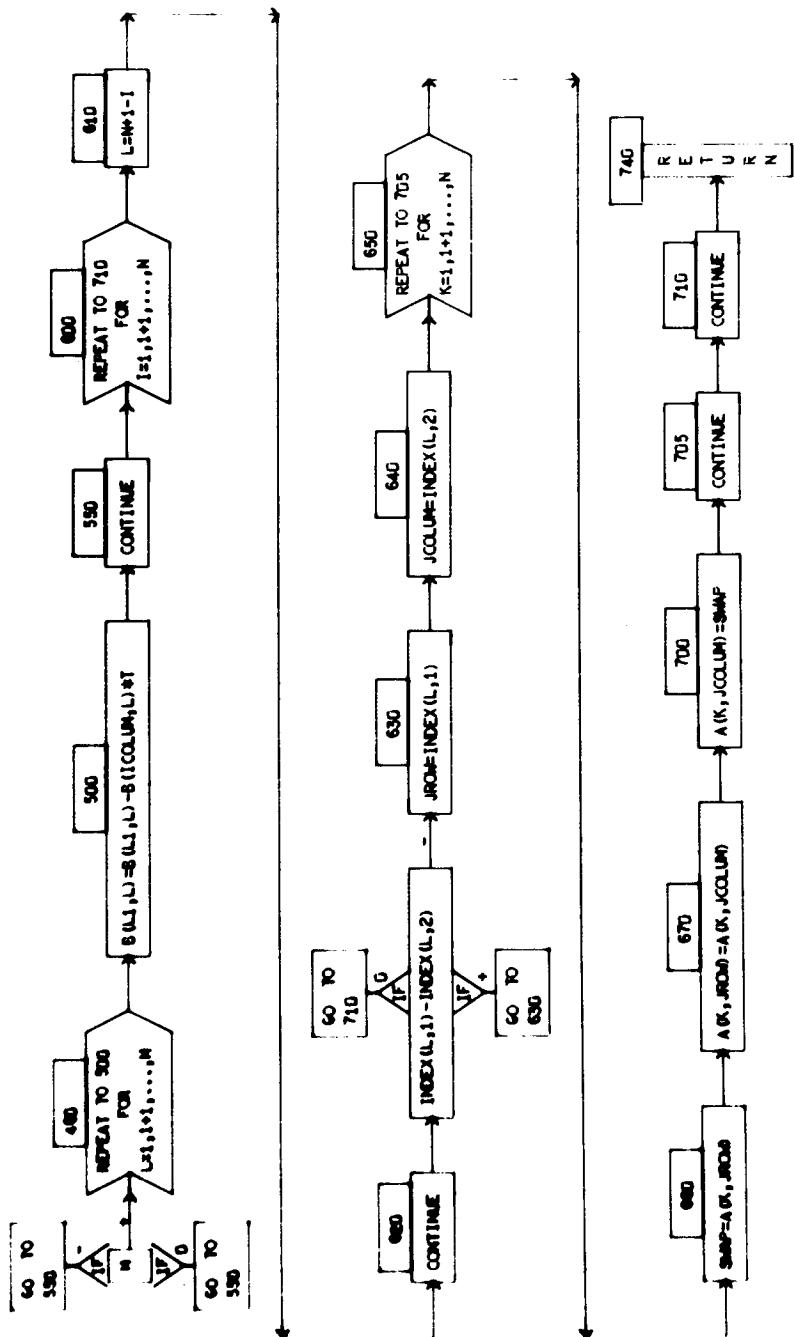
CIMENSIONEC VARIABLES					
SIMCOL	N	A	MAX,N	B	MAX,M
STORAGS					INDEX
SIMCOL					MAX,2
STORAGS					
SIMCOL					
STORAGS					
SIMCOL					

SUBROUTINE MATINV(A,M,N,CTERM,IPIVOT,INDEX,NMAX,ISCALE)









REFERENCES

1. Famili, J.; and Archer, R. R.: Finite Asymmetric Deformation of Shallow Spherical Shells. *J. AIAA*, vol. 3, no. 3, Mar. 1965, pp. 506-510.
2. Sanders, J. Lyell, Jr.: Nonlinear Theories for Thin Shells. *Quart. Appl. Math.*, vol. 21, no. 1, 1963, pp. 21-36.
3. Budiansky, B.; and Radkowsky, P.: Numerical Analysis of Unsymmetrical Bending of Shells of Revolution. *J. AIAA*, vol. 1, no. 8, Aug. 1963, pp. 1833-1842. (Discussion by Gilbert A. Greenbaum, *J. AIAA*, vol. 2, no. 3, Mar. 1964, pp. 590-592.)
4. Schaeffer, G.: Computer Program for Finite Difference Solutions of Shells of Revolution Under Asymmetric Loads. NASA TND 3926, May 1967.
5. Hoff, N. J.; Madsen, W. A.; and Mayers, J.: Postbuckling Equilibrium of Axially Compressed Circular Cylinder Shells. *J. AIAA*, vol. 4, no. 1, Jan. 1966, pp. 126-133.
6. Thurston, G. A.; and Freeland, M. A.: Buckling of Imperfect Cylinders Under Axial Compression. NASA CR-541, 1966.
7. Bushnell, D.: Axisymmetric Deformation of a Shallow Spherical Cap Clamped at the Edge and Submitted to Uniform External Pressure. LMSC-804536, Lockheed Missiles and Space Company, Palo Alto, Calif. Nov. 1964.
8. Bushnell, D.; and Madsen, W. A.: Machine Computation of Trigonometric Expansions. *J. EM*, Proc. ASCE, vol. 92, no. EM 6, Dec. 1966, pp. 157-174.

TABLE I. - IMPORTANT FORTRAN VARIABLES

FORTRAN Variable	Definition	Appears in Deck(s)
A(4, 4)	A matrix	6, ⑨, 10, 11
B(20)	B, b	③, 5, 7, 11, 12, 13, 15, 17, 18
BEE(4, 4)	B matrix	6, ⑨, 10, 11
BE1(10)	β at i - 1,	⑪, ⑫, 18, 19
BE2(10)	i, and i + 1	⑪, 18, 19
BE3(10)		11, ⑬, 15, ⑯, 17, 18, 19
BT1(10)	β_θ at i - 1,	⑪, ⑫, 18, 19
BT2(10)	i, and i + 1	⑪, 18, 19
BT3(10)		11, ⑬, 15, ⑯, 17, 18, 19
BX1(10)	β_s at i - 1,	⑪, ⑫, 18, 19
BX2(10)	i, and i + 1	⑪, 18, 19
BX3(10)		11, ⑬, 15, ⑯, 17, 18, 19
BXT1(10)	$\beta_{s\theta}$ at i - 1,	⑪, ⑫, 18, 19
BXT2(10)	i, and i + 1	⑪, 18, 19
BXT3(10)		11, ⑬, 15, ⑯, 17, 18, 19
C(4, 4)	C matrix	6, ⑨, 10, 11
CAPLL(4, 4)	$\bar{\Lambda}_K$, Λ_K	①, 6
CAPL1(4, 4)	$\bar{\Lambda}_1$, Λ_1	①, 6

○ Indicates the deck where the variable is defined

□ Indicates the deck where the variable is modified

△ Indicates the deck where the variable is both defined and modified

TABLE I. - IMPORTANT FORTRAN VARIABLES (Continued)

FORTRAN Variable	Definition	Appears in Deck(s)
D(20)	D, d	Ⓐ, 5, 7, 15, 17
DB(20)	$\frac{\partial B}{\partial s}, \frac{\partial b}{\partial \xi}$	Ⓐ, 5, 18
DD(20)	$\frac{\partial D}{\partial s}, \frac{\partial d}{\partial \xi}$	Ⓐ, 5
DEE(16, 20, 10)	$[B_1 - C_1 P_{1-1}]^{-1}$	⑩, 18
DEOMX(20, 10)	$\frac{\partial (\frac{1}{R_s})}{\partial s}, \frac{\partial w_s}{\partial \xi}$	Ⓐ, 5, 15
DMT(20, 10)	$\frac{\sigma_o h_o^3 (1-\nu)}{a} \frac{\partial m_T^{(n)}}{\partial s}, \frac{\partial m_T^{(n)}}{\partial \xi}$	Ⓐ, 18
DST(16, 20, 10)	$[B_1 - C_1 P_{1-1}]^{-1} C_1$	⑩, 18
DTT(20, 10)	$\sigma_o h_o (1-\nu) \frac{\partial t_T^{(n)}}{\partial s}, \frac{\partial t_T^{(n)}}{\partial \xi}$	Ⓐ, 18
E(4, 4)	E matrix	⑧, 9
ELL(4)	ℓ_K	①, 11
ELL1(4)	ℓ_1	①, 11
ET1(10)	η_θ at $i - 1$,	⑪, ⑫, 18, ⑯
ET2(10)	i , and $i + 1$	⑪, 18, ⑯
ET3(10)		⑪, ⑬, ⑭, 18, 19
ETT1(10)	$\eta_{\theta\theta}$ at $i - 1$	⑪, 18, ⑯
ETT2(10)	i , and $i + 1$	⑪, 18, ⑯
ETT3(10)		⑪, ⑭, 18, 19
ETX1(10)	$\eta_{\theta s}$ at $i - 1$	⑪, ⑫, 18, ⑯
ETX2(10)	i , and $i + 1$	⑪, 18, ⑯
ETX3(10)		⑪, ⑬, ⑭, 18, 19

TABLE I. - IMPORTANT FORTRAN VARIABLES (Continued)

FORTRAN Variable	Definition	Appears in Deck(s)
EX1(10)	η_s at $i - 1$, i , and $i + 1$	(11), (12), 18, 19
EX2(10)		(11), 18, 19
EX3(10)		11, (13), (17), 18, 19
EXT1(10)	$\eta_{s\theta}$ at $i - 1$	(11), 18, 19
EXT2(10)	i , and $i + 1$	(11), 18, 19
EXT3(10)		11, (17), 18, 19
EXX1(10)	η_{ss} at $i - 1$, i , and $i + 1$	(11), (12), 18, 19
EXX2(10)		(11), 18, 19
EXX3(10)		11, (13), (17), 18, 19
F(4, 4)	F matrix	8, 9
FFS(4, 10)	f_1 matrix	(11), 18
FLS(4)	f_K matrix	(11)
G(4, 4)	G matrix	8, 9
GAM(20)	$\frac{1}{r} \frac{\partial r}{\partial s}, \gamma$	2, 5, 7, 11, 16, 18
GEE(4)	g matrix	(18)
GEES(4, 10)	g_1 matrix	11, (18)
H(4, 4)	H matrix	6, (7)
ID(10, 10)	See description of subroutine MODES, Appendix E	14, 16, 17
IJS(10)	See description of subroutine MODES, Appendix E	14, 16, 17
IS(10, 10)	See description of subroutine MODES, Appendix E	14, 16, 17
J(4, 4)	J matrix	6, (7)
JD(10, 10)	See description of subroutine MODES, Appendix E	14, 16, 17

TABLE I. - IMPORTANT FORTRAN VARIABLES (Continued)

FORTRAN Variable	Definition	Appears in Deck(s)
JS(10, 10)	See description of subroutine MODES, Appendix E	14, 16, 17
MAXD(10)	See description of subroutine MODES, Appendix E	1, 14, 16, 17
MAXS(10)	See description of subroutine MODES, Appendix E	1, 14, 16, 17
MAXSY(10)	See description of subroutine MODES, Appendix E	1, 14, 16, 17
MT(20,10)	$\sigma_o h_o^3 (1-\nu) \frac{m}{a}$	4, 11, 15, 18
N(10)	n	④, 6, 7, 8, 14, 15, 16, 17, 18
OMEGL(4,4)	$\bar{\Omega}_K, \Omega_K$	①, 6
OMEGL(4,4)	$\bar{\Omega}_1, \Omega_1$	①, 6
OMT(20)	$\frac{1}{R_\theta}, \omega_\theta$	②, 5, 7, 16, 18
OMXI(20)	$\frac{1}{R_s}, \omega_s$	②, 5, 7, 12, 13, 16, 18
P(16, 20, 10)	P matrix	6, ⑩, 11
PHI(10)	Φ, φ	⑫, ⑬, 17
PHIT(10)	$\Phi_\theta, \varphi_\theta$	⑫, ⑬, 17
PHIX(10)	Φ_s, φ_s	⑫, ⑬, 17
PR(20, 10)	$\frac{\sigma_o h_o p}{a}^{(n)}, p^{(n)}$	④, 18
PT(20, 10)	$\frac{\sigma_o h_o p_\theta}{a}^{(n)}, p_\theta^{(n)}$	④, 18

TABLE I. - IMPORTANT FORTRAN VARIABLES (Concluded)

FORTRAN Variable	Definition	Appears in Deck(s)
PX(20, 10)	$\frac{\sigma_o h_o p_s^{(n)}}{a}$, $p_s^{(n)}$	4, 18
R(20)	r, ρ	2, 5, 7, 16, 18
TH(6)	θ	1, 15
TT(20, 10)	$\sigma_o h_o (1-\nu) t_T$, t_T	4, 11, 15, 17, 18
U1(10)	U, u at i - 1, i, and i + 1	(11), 15, 17
U2(10)		(11), 15, 16, 17
U3(10)		15, 16, 17
V1(10)		(11), 15, 16, 17
V2(10)	V, v at i - 1, i, and i + 1	(11), 15, 16, 17
V3(10)		15, 16, 17
W1(10)	W, w at i - 1, i, and i + 1	(11), 15, 16, 17
W2(10)		(11), 15, 16, 17
W3(10)		15, 16, 17
X(4, 20, 10)	x matrix	6, 11, 18
Z(4, 22, 10)	z matrix	1, (11), 12, 13, 15, 16